THE USENIX ASSOCIATION NEWSLETTER

# ;login:

FEBRUARY 1995 • VOLUME 20 • NUMBER 1

## CONTENTS

### FROM THE EDITOR

### USENIX NEWS

### SAGE NEWS

### FEATURES

### STANDARDS REPORTS

### BOOK REVIEWS

### ANNOUNCEMENTS & CALLS

Calls for Papers:

### ETC...

USENIX®

*;login:* is the official newsletter of the USENIX Association.

• *;login:* (ISSN 1044-6397) Volume 20, Number 1 (February 1995) is published bimonthly by the USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

• $24 of each member's annual dues is for an annual subscription to *;login:*. Subscriptions for nonmembers are $50 per year.

• Second-class postage paid at Berkeley, CA and additional offices.

• POSTMASTER: Send address changes to *;login:*, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley, CA 94710.

## Editorial Staff

- Rob Kolstad, Editor *<kolstad@usenix.org>*
- Bryan McDonald, SAGE Editor *<bigmac@erg.sri.com>*
- Nick Stoughton, Standards Report Editor *<nick@usenix.org>*
- Carolyn S. Carr, Managing Editor & Typesetter *<carolyn@usenix.org>*
- Monica Jensen, Copy Editor
- Eileen Curtis, Proofreader

## Membership and Publications

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, CA 94710
Telephone: 510/528-8649
FAX: 510/548-5738
Email: *<office@usenix.org>*

## Contributions Solicited

You are encouraged to contribute articles, book reviews, and announcements to *;login:*. Send them via email to *<login@usenix.org>* or through the postal system to the Association office. Send SAGE material to *<bigmac@erg.sri.com>*. The Association reserves the right to edit submitted material. Any reproduction of this newsletter in its entirety or in part requires the permission of the Association and the author(s).

*;login:* is produced with Framemaker 4.0 software provided by Frame Technology and displayed on an NCD X Terminal, donated by Network Computing Devices. The system is served by a Sun SPARCsystem 10 server. Output is sent to a 600 dpi QMS 860 laser printer, donated by Quality Micro Systems. The newsletter is printed on recycled paper.

# SAVE THESE DATES!

## Upcoming USENIX Events

### USENIX/SAGE at UniForum

**Tutorials on Internet Security, UNIX Power Tools, Firewalls, and The Law and Computers**

**March 12-13, 1995,** Dallas Convention Center, Dallas, TX

### 2nd USENIX Symposium on Mobile and Location-Independent Computing

**April 10-11, 1995,** Ann Arbor, Michigan
Program Chair: Jim Rees, CITI, University of Michigan
**Papers Due: March 6, 1995**

### 4th UNIX System Administration, Networking and Security Symposium (SANS IV)

**April 24-29, 1995,** Washington, D.C.
Sponsored by the Open Systems Conference Board and SAGE, the System Administrators Guild.
Program Chair: Rob Kolstad, BSDI, Inc.

### 5th USENIX UNIX Security Symposium

**June 5-7, 1995,** Marriott Hotel, Salt Lake City, Utah
Sponsored by USENIX, in cooperation with: The Computer Emergency Response Team (CERT), IFIP WG 11.4, and UniForum.
Program Chair: Fred Avolio, Trusted Information Systems, Inc.
**Abstracts Due: February 13, 1995; Authors Notified: March 8, 1995; Papers Due: May 1, 1995**

### USENIX Conference on Object-Oriented Technologies (COOTS)

**June 26-29, 1995,** Monterey Conference Center, Monterey, California
Program Chair: Vince Russo, Purdue University
**Abstracts Due: March 6, 1995; Authors Notified: April 3, 1995; Papers Due: May 15, 1995**

### Tcl/Tk Workshop '95

**July 6-8, 1995,** Royal York Hotel, Toronto, Canada
Sponsored by Unisys, Inc., and USENIX
Program Chairs: Ben Bederson, University of New Mexico and Will Wilbrink, Unisys, Inc.
**Papers and Demos Due: March 3, 1995; Papers and Demos Accepted: April 14, 1995; Camera-ready Due: May 23, 1995**

### USENIX Workshop on Electronic Commerce

**Summer 1995**
Program Chair: Dan Geer, Open Vision Technologies

### 9th USENIX Systems Administration Conference (LISA '95)

**September 18–22, 1995,** Monterey Conference Center, Monterey, CA
Co-sponsored by USENIX and SAGE, the System Administrators Guild.
Program Chairs: Tina Darmohray, Lawrence Livermore National Laboratory and Paul Evans, Synopsys, Inc.
**Abstracts Due: May 1, 1995; Papers Due: August 1, 1995**

For more information about USENIX and its events, access the USENIX Resource Center on the World Wide Web. The URL is *http://www.usenix.org*. Or you can send email to our mailserver at: *info@usenix.org*. Your message should contain the line: *send catalog*. A catalog will be returned to you.

FROM THE EDITOR

# Words

Remember the Queen of Hearts in *Alice in Wonderland*? She said something to the effect that "words mean precisely what I want them to mean: no more and no less."

Wouldn't that be a great way to run a lexicon? I think not.

I notice that the rhetoric filling our media (printed and electronic) seems to take perfectly good words and continues to twist them. I guess there's some notion that an idea will have more impact if you can fool people by using clever words.

By way of example, I was asked a few years ago if I ever stayed with a friend while between apartments or houses. Of course, the answer was yes. Timing that kind of a move is tricky unless you can come up with extra cash. Upon answering in the affirmative, I was told that I was just one member of the club of millions that have been homeless.

I don't think that was very helpful at any level. Furthermore, I thought that such tactics were fairly isolated. Regrettably, newspapers are now reporting some institute's findings with numbers based on similar questions. How useless.

My roommate just received another "hategram" from one of the local citizen's groups. It's full of quotes out of context and misinformation about "the opposition." As a propaganda piece, it's so-so. As another example of redefining words, it's outstanding.

Why do people do this? Even people whom I frequently deal with tend to change around the words they use to give the idea that things aren't as they really are. "I'm going cross-country skiing with a bunch of people," says one friend. Of course, as it turns out, he's skiing with a single companion and anticipates staying in huts with other individuals who happen to be on the same cross-country trails. Why didn't he just say it was a two-person effort?

I talk to potential customers on the phone frequently. They're all trying to figure out which weasel-words I'll use to get around telling them the real truth about my company's product. These consumers are trained to regard all claims with an incredible skepticism. Ain't that a drag?

I hope *;login:* doesn't follow this trend and that we can all resolve to communicate clearly in 1995 and beyond.

# Errata

Apologies to Scott Hazen Mueller, author of "Business Survival Skills 101: The Interview" in the December issue of *;login:*. We inadvertently omitted several paragraphs from his article. If you would like the complete version please send email to *login@usenix.org*.

CONFERENCES & SYMPOSIA

- Judith F. DesHarnais
  Conference Coordinator
  USENIX Conference Office
  22672 Lambert Street, Suite 613
  Lake Forest, CA 92630
  Telephone: 714/588-8649
  FAX: 714/588-9706
  Email: <*conference@usenix.org*>

- Peter Mui
  Vendor Displays
  Email: <*display@usenix.org*>

- Daniel V. Klein
  Tutorial Coordinator
  Telephone: 412/421-2332
  Email: <*dvk@usenix.org*>

WWW URL: *http://www.usenix.org*

AUTOMATIC INFORMATION SERVER

To receive information electronically about upcoming USENIX symposia & conferences finger <*info@usenix.org*> and you will be directed to the catalog which outlines all available information about USENIX services.

**The closing dates for the next two issues of *;login:* are February10 and April 10,1995.**

USENIX MEMBER BENEFITS

As a member of the USENIX Association, you receive the following benefits:

- Free subscription to *;login:*, technical features, system administration tips and techniques, international calendar of events, SAGE News, media reviews, Snitch Reports from the USENIX representative and others on various ANSI, IEEE, and ISO standards efforts, and much more
- Free subscription to *Computing Systems*, the refereed technical quarterly published with The MIT Press
- Discounts on registration fees for the large, multi-topic technical conference, the System Administration conference (LISA), and the various single-topic symposia addressing topics such as object-oriented technologies, security, operating systems, high-speed networking, and mobile computing – as many as seven technical meetings every year
- Discounts on proceedings from USENIX conferences and symposia and other technical publications. Discounts on the USENIX Association book published by The MIT Press. Now available: *The Evolution of C++: Language Design in the Marketplace of Ideas*, edited by Jim Waldo of Sun Microsystems Laboratories
- Discounts on BSDI, Inc. products. BSDI information: 800 800-4BSD
- Discounts on the five volume set of 4.4 BSD manuals plus CD published by O'Reilly & Associates and USENIX
- Savings (10-20%) on selected publications from McGraw-Hill, The MIT Press, Prentice Hall, John Wiley & Sons, O'Reilly and Associates, and UniForum
- Special subscription rates to *The LINUX Journal.* Phone: 206 527-3385
- Right to join SAGE, the System Administrators Guild

To become a member or receive information regarding your membership status or benefits, please contact <*office@usenix.org*>. telephone 510 528-8649.

# President's Letter

*by Steve Johnson*
*<scj@usenix.org>*

## Pentium in a Teapot

As I write this, Intel and IBM are firing salvos about the flawed Pentium chip, while Intel stock bobs wildly on the stock market, and our local paper has started a special section on Intel humor (Intel: where quality is job .9999978457246). There are a lot of lessons in this flap for all of us, no matter how it comes out. It is also interesting how the press has picked up on a few issues and totally missed others.

For any of you who may have been under a rock in December, or who have not paid attention to PC's because they were, well, PC's, here is a capsule summary. Last summer, Intel discovered that certain floating point divides could produce answers that were inaccurate in the fifth or sixth decimal place. They did a simple, back-of-the-envelope study that convinced them that the 'ordinary user' (presumably using WordPerfect and Excel) would encounter this problem once every 27,000 years, and, as a result of this study, did not tell anyone about this problem but continued to ship the flawed chip (2 to 4 million of them, depending on who you believe) while rolling a fix into later versions.

IBM (which has at least two other chips that they are selling besides the Pentium) announced that they stopped shipping the Pentium because their estimates suggested that the "ordinary user" would encounter this problem every 24 days. Note that this estimate differs from Intel's by a mere factor of 400,000 or so (isn't science wonderful?). I turn eagerly to the paper each day to see the latest bombs go off.

In all this press coverage, I think there are two issues that were almost ignored. Intel made much of the fact that this error is only in the fifth decimal place, where it will not affect most graphics and financial computations. However, the division is probably not the last thing done in the computation. In some cases, the error might affect the drawing of a graph where the error would be unnoticeable. In other cases, the error might affect a control circuit and cause it to fail to reach equilibrium. In some cases, these division errors might cause series to diverge. In other cases, small errors might, through addition and subtraction, lead to total loss of accuracy. Once the error happens, the effect on the final program is impossible to generalize, but can be catastrophic.

As we are all well aware, computers make mistakes, and software is probably more to blame these days than hardware, so why has this problem made all the headlines? Here I think there are three factors: Intel has aggressively marketed their chip ("Intel Inside", etc.), the problem happens silently, and, the most important factor, Intel didn't tell anyone.

I think the press largely missed these last two points. While it is true that most CPU's in my experience have had bugs, these have mostly involved the chip going into states where it is locked up, ignored interrupts, or lost state, leading pretty directly to a system crash or panic. In fact, most software errors have a similar property – they announce their presence loudly and clearly. As unpleasant as a system crash is, it is vastly better than, for example, bad file system code that trashes your information without warning. It is almost unendurable to work on a computer with a flaky file system: users trample each other in their rush for the exit. And the Intel bug is of that type: data is produced that is incorrect, and no indication of the incorrect data is provided.

But it is the last point, Intel not telling anyone, that has really generated a lot of the fuel for this fire. By withholding this information, Intel appears to be somewhere between paternalistic and arrogant (let's not tell the poor dears about this, it will only upset them. . .); the truly paranoic see Intel as having understood the seriousness of the problem and engaged in a Pentigate cover-up. Intel's initial unwillingness to replace flawed chips on request rubbed salt into the wound. It is now clear that Intel's handling of this issue became a bigger problem than the bug in the divide instruction.

So why are we holding Intel to such a high standard? Many UNIX systems have been far less reliable than the Pentium. I think the answer is in those eternal truths of marketing: put the customer's needs first, communicate openly with them, and put your whole company behind making them happy. Intel apparently missed on all three counts.

UNIX was designed for programmers by programmers, so, at least in the early days, we met the customers and they were us. The UNIX community has always communicated openly and often, both electronically and through organizations such as USENIX. And many UNIX vendors have put a huge amount of money and time into developing UNIX and its descendants into mature, highly reliable systems. We have not had a UNIgate, although we have certainly had occasional (and short-lived!) unreliable products. But our community has worked with those vendors who would talk with us, and blown the others away.

If Intel is serious about entering the workstation market, it will need to learn how to get along in our environment, and there are signs that they have begun to do so. They are working with a top-flight group of independent industry and academic people to characterize the bug and develop a software workaround for it (People who want more technical information might enjoy accessing the Pentium papers at *http://www.mathworks.com*). They are developing a net presence. We hope they have learned from this experience. We can learn from it too.

# Very High Level Languages Symposium Report

## October 26-28, 1994

*by Jeff Haemer*
*<jsh@canary.com>*

This was my first "little" conference. I loved it.

I look forward going to the regular Summer and Winter conferences, but the smaller conferences always seemed too specialized for me, and I never thought about going. This one, however, was in Santa Fe, New Mexico – practically next door to Boulder, Colorado, where I live, – so I just drove down. Took me about eight hours, but Evi Nemeth, who also drove, tells me that if you drive fast enough, you can both make up for time that you lose by running out of gas *and* make up for the lower cost of the little conferences with speeding fines.

So, how are little conferences different? Well, first of all, smaller conferences fit in smaller places. Instead of spending a week in a mega-hotel with big chandeliers and projection TV screens in Dallas or Boston, I got to spend three days in sunny Santa Fe, the second oldest city in the U. S. (I hasten to say that New Orleans, where the large January conference will be held, is also a wonderful place to visit.) Plus, it's high. Not only was Santa Fe founded before the Pilgrims landed at Plymouth Rock, it was founded about 7000 feet (2134 meters) farther up. Even Coloradans huffed and puffed a bit.

But for me, the most interesting surprise was that a little conference allows you to hear everything. Unlike larger conferences, there was only one track; the technical papers were interleaved with invited talks. And since there were about 30 authors, out of 150 attendees, chances were fair that you would end up at lunch or dinner with someone who had a paper you wanted to know more about.

Having said all these good things, I have one complaint: I think the conference was mis-named. When I hear "very high level languages," I think of Russian. This conference was about languages like Perl, Python, and REXX. Still, if you want to hear about those, read on: I'll tell you why Jon Bentley loves Visual BASIC, and how I won a free Perl 5 tutorial because Richard Stallman hates Tcl. Tom Christiansen, the conference chair, began by announcing the now-traditional, conference contest. This one was to describe a language that you'd like to see, one that you think should have been written, or one that you think shouldn't have been written.

This led immediately into Jon Bentley's keynote, "Languages I've Loved." If you've read any of Jon's *Programming Pearls* columns or books, you'll know that Jon is attracted to little languages. He began by showing us that he sometimes means *really* little languages.

In 1984, when Kernighan and Pike's *The Programming Environment* was first published, I bought a copy and began working my way through the book, typing in every example and trying to understand what was going on. Chapter 8, "Program Development," is 54 pages of C, yacc, lex, make, and other tools all combined in interesting ways to build a sophisticated desk-calculator language, called hoc. My fingers got tired, and I still haven't finished the chapter.

Jon gave me an alternative, which he calls hawk: hoc in awk. Here's his initial implementation:

```
{
        f = "hawk.tmp"
        print "BEGIN { print " $0 ")" > f
        close(f)
        system("awk -f" f)
}
```

Notice that hoc offers all the familiar awk functions and functionality, but pares away some of awk's syntactic overhead to provide a simple desk-calculator. A more sophisticated, final version, which he also showed, is 27 lines long and still fits on a single overhead.

From there, Jon went on to talk about a variety of other useful languages, ranging from pic preprocessors like grap and chem, to big languages like PRL, which describes integrity constraints for databases and is used on 5ESS phone switches (60% of all telephone subscriber lines talk to 5ESS's). Each language, however, illustrated lessons that he first brought out in hawk:

- start simple
- try to let an intermediate language, like C or awk or pic, do all the hard work,
- if you want people to use what you've written, provide a zoo of "Kernighanian" examples, with practical problems, good style, and useful idioms.

Jon wound up by showing his current favorite language: Visual BASIC, which he says he's grown tired of defending. He says that, so far, he's only lost two arguments with his fourteen year-old son. The first was whether or not to buy a color printer (Jon had color overheads), and the second was about using Visual BASIC. With a $946 computer, a $13 sound card and speaker, he demonstrated a pair of multimedia applications that he'd written the evening before: a beautiful sort animation, and "Oust-a-matic," an inexpensive simulation of John Ousterhout – perfect for people who can't get to John's Tcl/Tk tutorials at USENIX.

I thought about Dijkstra's pronouncement that anyone who learns BASIC is ruined as a programmer for life, and wondered what he'd say about the new generation of programmers who will learn multimedia programming using Visual BASIC.

In response to Evi Nemeth's question, "What language would you teach to college freshmen?" Jon replied, without hesitation, "English."

Following a break, we got a language-taster party. The rest of the day there were language overviews of Perl, from Larry Wall; Tcl/Tk, from John Ousterhout; Python, from Guido van Rossum; REXX, from Pamela Taylor; and Icon, from Clint Jeffery.

Of these, the two poles seemed to be Tcl/Tk and Perl. John Ousterhout's motto is "Cleanliness is next to Godliness." In Tcl, even conditionals and loops are functions. Larry Wall, in contrast, has a linguistics background; for him, simplicity and uniformity are nice when they're convenient, but expressiveness is primary. To Dave Korn's offhand comment about Perl's having everything thrown in, including the kitchen sink, Larry smiled and riposted, "Does your house have a kitchen sink?"

Besides a quick Perl overview, Larry Wall gave us a tour of the new features in Perl 5. The two most startling developments are the addition of objects, and the subtraction of a lot of complexity. The yacc grammar for Perl 5 is half the size, and the reserved words are cut by two-thirds. Despite this simplification, there now are three ways to do inheritance and four ways to invoke a method.

This polymorphism of polymorphisms is more-or-less what you'd expect from Perl, and provoked some audience discussion of the disadvantages of Perl's theme: "There's more than one way to do it." Larry commented that he now realizes one of those ways is "If you don't like all Perl's flexibility, don't do it in Perl."

I'd seen John Ousterhout talk several times before, so his overview of Tcl/Tk was good, but not new to me. What I did find interesting were some social details. First, almost every other speaker contrasted his language with Tcl. Not one speaker contrasted his language with COBOL or LISP.

Indeed, Tcl's popularity has become so widespread, that Richard Stallman recently published a manifesto urging people not to use Tcl, arguing that the language is deeply flawed, and that widespread use of Tcl would hinder adoption of a "truly good" scripting language. John made reference, in his talk, to the "Free Software Fundamentalists."

(Personally, I guarantee that the FSF could displace Tcl almost overnight by distributing a clone of Visual BASIC. What's Bill Gates going to do – sue the FSF for everything it owns? :-)

Second, John noted that Tcl's popularity comes from add-on packages to Tcl, particularly the Tk extensions and Don Libes' Expect. Several other talks at the conference also stressed how important it is to make it easy to produce and use libraries, special-purpose packages, and optional add-on modules. Closed languages, like PASCAL, are not in vogue.

Finally, John, unlike almost every other speaker, didn't announce any plans to make his language object-oriented. In the evening, Steve Johnson, in his "Objecting to Objects" talk, argued that this isn't necessarily bad. "Messages," Steve says, "are the *goto*'s of the 90's."

Be that as it may, Python is unapologetically object-oriented. Like most of the other languages presented throughout the conference, it is also interpreted, embeddable, good for rapid prototyping, available for free over the net, and has many libraries and a large user community. Python is said to be somewhat Perl-like, but better suited to programming in the large. Python is expected to see heavy use in the web, and there has already been one NIST workshop this year to help plan Python's evolution. There may be another in the Spring.

Like Perl, Python's performance is said to be good, though Guido sidestepped performance questions with an overhead showing that Python runs unsurprisingly faster on an SGI Irix machine than it does on a Macintosh.

There is little question that the syntax of Python is better than Perl's (or Tcl's, for my money). It's long been in vogue to say that only semantics are really important, and to refer to syntax with either apology or disdain. In German, "Sytacticsugar" is probably already a word. Next time you hear someone slight syntax, ask him to do long division in Roman numerals, or calculus with Newton's notation instead of Leibnitz's.

Two particularly telling examples of the importance of syntax are the popularity of Dave Korn's ksh, which gave us an elegant way to say "repeat the last command," and Aho, Weinberger, and Kernighan's awk which gave us a powerful pattern-scanning and processing language with C's syntax. Don't believe me? Quick, what does this awk code fragment do?

```
for (i=0; i<100; i++) printf("%d\n", i);
```

Now, quick, how would you write that in Tcl or Perl? When John Ousterhout explained that one of Tcl's advantages was a syntax similar to sh, C, and LISP, he illustrated his point with an assignment statement: set a 47.

Since I actually think syntax is important, I sat right up when someone asked Guido to contrast a Python feature with a similar feature in awk, and he replied that he had never used awk. He wasn't even unusual in this. Although it was one of Jon Bentley's "loved languages," a surprisingly large number of the language authors and wizards in the room admitted, by a show of hands, that they, too, had never used it. I can, without hesitation, give Clint Jeffery the best-talk-about-a-language-I'll-never-use award. It's sad but interesting.

A little background is in order. Clint is Ralph Griswold's most recent Ph.D. graduate. Everything Griswold writes is worth reading. I haven't had access to a SNOBOL compiler since about 1981, but Griswold, Poage, and Polansky's *The SNOBOL4 Programming Language* is still on my bookshelf. The best article on porting software I know is his 1977 (!) article, "Engineering for Portability," which is also on my shelf in a book of otherwise completely dull and hopelessly antiquated papers on the subject.

Once he gets away from English, though, Griswold's taste in languages is both powerful and odd. In case you haven't ever programmed in SNOBOL, here's a program to reverse strings:

```
      reverse =
rem   text len(1).c =   :f(done)
      reverse = c reverse:(rem)
done
```

Icon is both more normal and odder. Its syntax is PASCAL-like, but compared to other languages I know, it is from semantic Never-Never Land. For example, Icon has both *generators* and *goal-directed* evaluation. Generators are expressions that can produce multiple results, and goal-directed evaluation means results are computed only when required by the surrounding expressions. Try this on for size:

```
if 12 < !L < 20
      write("teenager(s) present")
else
      write("no teenagers")
```

The expression *!L* is a generator that generates the list of ages held by *L*. The *if* iterates through the list, looking at each value generated by !L in turn, until something succeeds. If the generator runs out of values, the expression fails.

Icon is free, powerful, and fast, and it has great string and structure-processing features. Associative arrays can have *any* data type as an index. Icon is good for complex algorithms, especially ones that need backtracking.

So if Icon is so wonderful, why won't I use it? Candidly, it is syntax again. Icon's syntax is different enough from C that I'd have to think hard about both new syntax and new semantics. My rule is that I pick up new tools only when the overhead of learning them is a small part of solving the overall problem. If I have to do a large AI project, I'll undoubtedly reconsider, but for now, Icon's activation energy is too high.

That confessed, the talk was both fun and insightful, and I at least know what sorts of projects I might use Icon for. After I learned about Tcl and Perl, a couple of years elapsed before I actually used them, too. (Incidentally, I have two spellings for his last name in my notes – "Jeffrey" and "Jeffery" – I've picked "Jeffery," but if I blew the coin flip, Clint, I apologize.)

Pam Taylor's REXX talk was surely the one that took the most nerve to give. First of all, as she pointed out, because of REXX's age (the first specs were from the 1970s) and its origins (IBM mainframes), REXX lacks one of the most important features of a Very High Level language: dollar signs. I don't mean it's not economically important; I mean that assignment statements don't look like this: `x=$y`. How do REXX programmers ever get by?

Once you get past this mysterious omission, and a few other oddities – as in PL/1, there are no reserved words; keywords are only keywords when used in the right context – REXX has some of the same advantages as Tcl. It's interpreted, embeddable, and extensible, Lotus Ami-PRO, for example, uses REXX as an embedded language.

There's even an object-oriented REXX of unknown name coming. (It was called ORYX for a while, but someone else has a trademark on that name. REXX was originally called "REX," but got a second "X" for the same reason.)

REXX is good for rapid prototyping but powerful enough for mission critical applications. It has a rich set of instructions and functions, and it runs on everything from Commodore Amigas to IBM mainframes.

Although REXX began in the IBM world, the specification is public and the only trademarks and copyrights are on a font commonly used for the word "REXX," and a King-of-Spades logo, frequently seen on documentation. Indeed, except for ksh, it was the only language presented at the conference for which there's a standards effort – X3J18. (I'm interested in standards, so this last item caught my personal attention.)

The REXX world and the UNIX world haven't overlapped much until now, but that's changing. There are already versions of REXX which supply functions like _stat and _gethostbyname. Mike Cowlishaw, REXX's author, and Larry Wall have modified their languages after conversations with one another. There is already a comp.lang.rexx, and folks at the conference were asking, "Is there a Tk/REXX yet?" (Actually, when someone asked this question, voices from several parts of the room cried out in gleeful unison, "T. rexx!")

There were occasionally funny culture clashes. After Pam ran one demonstration as a canned executable, someone from the audience asked "Could you type that program instead?" "Certainly," she replied, and entered `TYPE SAMPLE.BAT` to display it on-screen.

After dinner, there were three talks, which I had to miss. Two of them were repeats of popular invited talks by Dan Klein and Steve Johnson from earlier conferences, and one was a technical paper on an implicitly parallel composition language by R. Jagannathan and Chris Dodd.

Day two started out with another language overview, this one on ML (for "metalanguage"), by Andrew Koenig. In addition to being a well-known C++ guru, Andy is an ML user and fan.

ML is (mostly) a functional programming language. Although ML is strongly typed, it figures out all data types from context. There are no implicit data conversions, and no type declarations, either. It is also the only programming language that I can think of with no assignment statements.

While he isn't put off by some of ML's unusual properties, Andy admits that it is the twenty-first programming language he's learned. I didn't have the impression that he was the norm. Neither, I think, did he.

Early on, he warned us that "ML makes far fewer concessions to practicality than you're used to, but the payback is occasional breathtaking elegance." True enough, but it wasn't long before John Ousterhout asked whether it would be unfair to characterize ML as a language for people with excess IQ points. Someone in the audience volunteered that ML is a standard undergraduate programming language at Cambridge, without saying whether that meant "yes" or "no."

Here's a sample overhead: Here is a clausal definition of the `length` function:

```
- fun length nil = 0
=       | length (_::t) = length t + 1;
val length = fn : 'a list -> int
```

We can use _ as a "name" when we know we're not going to need it; this allows us to avoid having constantly to think up new names we don't intend to use. The talk was frequently punctuated with the phrase "Trust me."

The ML compiler is robust, and fairly portable. (For example, it won't run on Windows 3.1, but will run on NT.) The compiler itself is very slow, but generated code is quite fast. Although it's about the same age as BCPL, the current version, freely available by anonymous ftp from *research.att .com:dist/ml*, is 0.93. (Version 1.0 is projected to appear in 1996.)

Following a break, we had a talk by Andrew Koenig about ML. No, really. This one was a technical paper, called "An Anecdote About ML Type Inference," illustrating how ML's strong typing uncovers programming errors in unexpected ways. Since no one else knew ML, it gave us a chance to ask more ML questions, which we needed to do.

The examples Andy used in the first talk were factorials and Fibbonacci series. In this talk, he "wanted to illustrate writing something useful," so he went through a merge sort. At first, I couldn't figure out why this was more useful. I don't currently have a utility that generates factorials or Fibbonacci series, but every system (and every .2 system, or not) has sort(1). Then I remembered, in his first talk, Andy had mentioned that ML ports to Windows/NT.

In another concession to practicality, he revealed that ML supports arrays – important Andy said, because, "I can't think of any programmer who hasn't asked his boss for arrays."

The rest of the morning was taken up with Scheme. Brent Benson led off with a talk about libscheme: Scheme as a C library. Libscheme provides an embeddable language, like Tcl, but with LISP syntax, which Brent happens to prefer. It's a from-scratch implementation of Scheme, and carries no copyleft. The talk was relatively uncontroversial.

Luckily, this was followed by Adam Sah and Jon Blow's talk, which had two basic themes:

- If you already have an implementation of a VHLL, you should throw it out and re-write it in Scheme, and

- Once you re-write it, throw it out and use Scheme instead.

To underscore the second point, Adam trotted out "bugs" in several popular VHLLs, reassuring us that what he said was technically correct, because he was an expert in each of these languages. There was frequent head shaking from the languages' authors, who were in the audience. Adam's entire delivery was at warp New York, and he interrupted nearly all questions before they were done.

For me, the Question-and-Answer period for this talk was one of the highlights of the conference. Tom Christiansen joined in the spirit, leading off with "Everything you said about Perl was either misleading or bogus," and then talking just as fast as Adam, just as loud as Adam, at the same time as Adam.

After this impressive burst, Andy Koenig, the session chair, suggested that further discussion be deferred to a Vultures-of-a-Feather session. In the early afternoon, we heard a lot about Tcl and Tk.

Karin Petersen started us off with a lovely talk about the design challenges of trying to use Tcl/Tk to write applications for a personal digital assistant, with IR communication capabilities, a 20x8 screen, and no keyboard. Applications included a remote-control manager for slide presentations and for a Canon video camera; a context-aware reminder application that reminds you to do things based on what room you're in; and a remote post-it note editor, which lets you jot down memos in a meeting that pop up on your workstation as post-it notes. She warned us that she may come to the next conference with a talk about writing Tcl applications for a Timex digital watch with an IR receiver.

Following Karin, Chris Lindblad talked about using Tcl to control a participatory multimedia programming environment. What that means is writing programs that actually watch TV, so you don't have to watch it yourself. For example, chances are good that you've seen programs like `puzzle`, from the X11r5 distribution, that will chop a picture into squares, make a 15-puzzle (a "boss" puzzle) out of the squares, and then either let you try to solve it yourself, or solve it for you. Chris has code that will do the same trick when the image being chopped up and scrambled is coming from a movie or TV screen, so that all the pieces are continuously changing as you, or the computer, push them around.

Wrapping up this session, Malcolm Beattie talked about "TkPerl: Mutant Child of Deformed Parents." The work is available, but still in alpha test, and relies heavily on features of Perl 5 – itself still a neonate. Tantalizingly, Malcolm's goal is not to make it easier to write Perl instead of Tcl, but to make it easier to write Perl instead of C.

I missed large chunks of the talks in the next session for a variety of reasons. I regret this. Unless the organizers conspired to put all the bad talks into one session, I assume that all three talks were of the same high quality as those in the rest of the conference. Andy Schorr gave a talk about Programmed Graph Re-writing Systems (PROGRES), John Snyder gave a talk about building end-user Systems at Bell Labs, and Steve Johnson talked about Matlab.

Finally, just before the reception, Dick Dunn recovered from food poisoning in time to give an invited talk on PostScript. PostScript, Dick explained, is FORTH on bad drugs. Dick's opening example, which drew applause, was

```
%!
/Times-Roman findfont 12 scalefont setfont
72 720 moveto (hello, world) show showpage
```

(Trivia buffs take note: November 21 is "World hello day.")

The most interesting and most frustrating talk was Dave Korn's, on Friday morning. Most folks take the shell too much for granted. It is, I believe, the most powerful and most intellectually interesting programming language on my computer. This was true even before Dave Korn; as early as the Mashey shell, studies showed that over 80% of the executables on large installations were shell scripts. Ksh, however, with its command-history mechanism, has changed the way I interact with computers. I could do a lot of work on a machine without a C compiler before I felt frustrated. I can't say the same about a machine without a conforming shell.

This makes ksh-93 really exciting. The first major release in five years, ksh-93 contains many of the useful features of Tcl and Perl, including floating point arithmetic, associative arrays, full extended regular expressions, a hierarchical name space for shell variables, Tcl-like embeddability, a partial compiler, and active variables. (You can, for example, define the variable in a way that makes always contain the current date.)

Unfortunately, in contrast to nearly every other language presented at the conference, ksh isn't freely available. Bell Labs has decided to become the IBM of the 90's – not noticing, one assumes, that IBM is trying as hard as it can to *stop* being the IBM of the 90's.

Dave's talk was followed by Storrs Hall's talk on Fornax, a language he named after a Southern Hemisphere constellation "both because Fornax (the furnace) is a place where you can rapidly forge tools, and it is very obscure." Like ML, Fornax is for someone who likes playing with brain-twisting syntax and semantics. Its fundamental data type is n-dimensional data aggregates, which it manipulates with a syntax stolen from APL and SNOBOL. The language actually requires, unapologetically, that you put a comment between function definitions.

Refreshingly, the author isn't looking for converts. He has written a language that he enjoys, and is having a heck of a good time with it. He conveyed that sense of fun convincingly. To the question "What applications are available in Fornax?" the author answered, cheerfully, "None. You can't get it." Pity. Maybe sometime in the future.

Ending this session was another outstanding Icon talk by Clint Jeffery, who explained that his goal was to make graphics in Icon "as gosh-darn totally trivial as possible." This he did, not by adding windowing, but by looking for simple, elegant ways to add powerful graphics. This wasn't Icon/Tk for building menu-driven windowing applications, it was research into how to design a language for writing virtual reality applications.

Two talks targeted at the real world were presented in Session 5. In the first, David Ladd and Christopher Ramming gave some details about PRL, a language to help populate the databases of 5ESS switches ("a database that occasionally switches phone calls"). Their most thought-provoking conclusion was that it is sometimes worth restricting the power of a language in order to make the programs easier to analyze. In the second, Gary Pollice talked about his graduate research on languages to produce scanners for C and C++ without a zoo of ifdefs.

After lunch, R. Stockton Gaines talked about "Dixie," an unimplemented idea, and Phong Vo gave a talk on feature-based portability that had some brilliant individual points, but that I had trouble following because of an unfortunate combination of the speaker's accent and the sound system.

Steve Johnson wound up the day with a somewhat pessimistic footnote talk about declining software productivity measured in bang for the buck, and argued that VHLLs were a way around that problem.

Good conference. I learned a lot, and next time I have a chance to go to a "little" conference, I'm going.

For those of you who want more details about this one, the technical papers are, as always, available in bound form from the USENIX Association. As at regular conferences, the invited talks were only distributed as overheads, but most were also (audio) taped, so if you're particularly interested in details from one of the invited talks, contact the Association.

Oh, I almost forgot. I'm not a languages wizard, so whenever I got too lost by arguments about reference counts versus garbage collection or lambda expressions, continuations, and closures, I'd amuse myself by writing entries for the contest. At the wrap-up, Tom Christiansen announced that I had won a free Perl 5 tutorial with:

> OverReac/Tk – a toolkit for writing applications that push *your* buttons

and

> WayOverReac/Tk – an implementation of OverReac/Tk from the FSF.

# Report on the First Symposium on Operating Systems Design and Implementation

*by Dave Mitchell*
*<dwm@orca.com>*

The first OSDI Symposium was held at the Monterey Conference Center in Monterey, CA on November 14-17, 1994. This three-in-one symposium was sponsored jointly by USENIX, ACM-SIGOPS, and IEEE-TCOS, replacing three earlier conferences on operating systems: the Mach Symposium, Microkernels and Other Architectures, and SEDMS.

The symposium was well attended, with 370 people registered. Monday consisted of six half-day tutorials, covering the architecture and internals of Spring, the GNU Hurd, and Chorus, as well as the use of Isis and Horus for distributed computing, the x-kernel for networking, and distributed shared memory for parallel computing. Out of 178 submissions, 21 papers were chosen to fill the bulk of the next three days of technical presentations. A workshop on Mach and Chorus was held the last afternoon, and a variety of BOF and works-in-progress sessions filled every spare moment in between. The conference was well-received and fast-paced, driven by the energy and enthusiasm of Jay Lepreau, the program chair from the University of Utah.

## Tuesday, November 15

David Patterson (University of California Berkeley) gave a keynote address that was full of good advice and hilarious in its sarcastic, backward approach: "How to Have a Bad Career in Research/Academia." Running through a list of bad career moves, he first advocated becoming THE leading expert in a field by inventing a new one whose payoff is at least 20 years away so you can use it to support most of your career. This allows you to avoid working with others so you won't have to share the credit (a prima donna personality is helpful). For an example, Patterson offered up the insight that computer security is increasing in importance, and capability-based systems got the focus nearly right; a useful new idea would be to create a list of all the things a process *cannot* do.

The key initial research question of disability-based systems would, of course, be whether you should store disabilities with each user, or with the objects that they can't access. Encrypted disabilities and disability-based addressing were further fertile topics. Another area ripe for investigation in the OS arena was "Omni-Femtokernels" – omnipresent in everything from VCRs to automobiles, and femto, as in a micro-kernel only more so. Its key contribution would be to provide employment to OS researchers when everybody is running DOS.

Patterson's next bad tip was to let complexity be your guide. This will confuse your competitors, make it easier for you to claim credit for subsequent good ideas, and easier to publish the N+1st incremental change, since nobody else will understand your points well enough to criticize them. Further advice for a bad career included several ways to avoid being proved wrong: uphold the OS tradition of promising "it will be working soon" to avoid actually implementing, avoid benchmarks and quantitative experiments in favor of hand-waving intuitive positions, and choose projects with payoffs of more than 20 years, since that will give you 19 safe years.

This advice continued on, noting that avoiding feedback is a great way to keep from being distracted by others, and stating that publishing papers is technology transfer. Since publishing is the key to success, legally changing your name to Aaaanderson will keep you at the top of everybody's reference list, and the LPU (least publishable unit) is your friend: one good idea spawns four journal papers, which beget 16 extended abstracts, and then 64 tech reports. The first half of the keynote finished by going over some tips for bad writing, giving bad talks, and making bad slides (this last section was cause for more than a little giggling in the audience during later presentations).

For the second part of the keynote, Patterson outlined what he recommended as a positive approach. His primary goal is to aim to have an impact, and having many short projects gives more chances for impact, as does choosing to focus on "real stuff," or problems that somebody cares about. Getting feedback is key; seeking critics provides a valuable reality check. Technology transfer requires convincing others, and the missionary work required extends far beyond merely publishing, since industry is reluctant to change and you need one bold company to both take a chance on your idea *and* be successful. This might seem like a large portion of motherhood and apple pie, but his plug for quantitative scientific methods instead of varying everything at once got a definite and positive response from the audience.

Carl A. Waldspurger gave the first talk, "Lottery Scheduling: Flexible Proportional-Share Resource Management," which was awarded Best Paper by the program committee, and also prompted a large amount of hallway conversation. This work was motivated by the desire to provide flexible, responsive control over service rates in multithreaded systems, so that quality of service guarantees could be provided to applications ranging from long-running background computations through interactive applications to networked multimedia displays requiring timely response. Most schedulers in use today rely on the notion of priority, either usagedecayed or fixed. The assignment of these priorities and adjustment policies are poorly understood and largely ad hoc.

Existing fair-share schedulers are typically coarse-grained and attempt to mediate resource usage among groups of processes. They fail to provide the dynamic, rapid control of scheduling decisions, at a time scale of fractions of a second, that is needed to manage a range of applications with conflicting requirements. Lottery scheduling is a randomized mechanism that implements proportional share resource management, where the consumption rates of active processes are proportional to the relative number of shares they are allocated. At the time of each scheduling decision, a random number within the range of allocated tickets is chosen, and the holder of the corresponding ticket is run. Thus the probability that a given task will be scheduled is proportional to the number of tickets it holds, and equal to that number divided by the total number of active tickets. Any task with a non-zero number of tickets will eventually win a lottery, so starvation isn't a problem.

Since the probabilistic aspects of random lotteries are well understood, it is easy to reason about the behaviors of this method of scheduling. The paper discusses the details of implementation, the room remaining for optimization, and briefly discusses applying this technique to non-CPU resources such as mutex locking. These lottery tickets can also be temporarily transferred during various blocking operations, allowing servers to provide good response without having any tickets of their own. This method also prevents priority inversion problems by increasing the probability of running a piece of code holding a mutex with pending waiters. Somebody with a real-time background raised the issue of the lack of a guaranteed bound on scheduling variance in the face of hard real-time requirements; this was countered with a response considered heretical by hard-real-time folks, "CPUs are getting faster, we can decrease the scheduling quantum."

The second talk, by Brent Welch (Xerox PARC) was "Scheduling for Reduced CPU Energy." He discussed ways to decrease the amount of energy used by battery-operated computers which go beyond simply spinning the disk down during idle periods. That method is necessarily coarse grained, given the spin-up latencies of disks. The energy consumption of a chip is proportional to the square of the voltage used. Chips can often be made to run at lower voltages, but this increases settling time, requiring slower clocking. The potential benefits come from noticing that, with a non-zero amount of idle time and perfect knowledge of the future, energy could be saved by turning the voltage and clock down so that the computation was stretched out to fill the time which would otherwise have been idle, running the same number of useful cycles, but at a lower total energy cost due to the non-linear relationship between CPU speed and power consumption. A number of algorithms were examined and compared to the best savings possible given perfect knowledge of the future.

Fred Douglis gave the last talk of the morning, "Storage Alternatives for Mobile Computers." His group used hardware measurements and trace-driven simulation to survey the trade-offs between magnetic disk, flash memory cards, and flash disk (flash memory with a disk-like interface). Disk consumes an order of magnitude more energy even with active power management, while flash memory (with either interface) offers low energy consumption, good read performance, and acceptable write performance. The surprise in this paper was learning just how much worse flash memory performs at higher levels of utilization, where the need for erasure drastically impacts both write throughput and energy consumption.

After lunch, the first three talks concerned file systems. Liuba Shrira (MIT) started off the session by presenting "Opportunistic Log: Efficient Installation Reads in a Reliable Storage Server." In a distributed store, client caches can out-perform page-based caches when organized as caches of objects smaller than a page in size. The problem with this approach is that object servers must then perform additional disk reads to install modified objects into their containing pages. The opportunistic log is essentially a large object cache that reduces disk traffic by removing the installation reads from the commit path, and scheduling bulk updates more efficiently (further object fetches hit in the modified object cache). Opportunistic logging allows an object server to outperform a page server while using a much smaller cache than would otherwise be necessary.

Gregory R. Ganger (University of Michigan) presented "Metadata Update Performance in File Systems." The focus of this paper was on solving the problems caused by the typical UNIX practice of maintaining file system integrity by performing metadata updates synchronously. Conversion to a log-based file system is one option, while other systems use asynchronous writes and pass the write ordering requirements down to the disk scheduler. This latter group can outperform the more conventional approach by a third, but are still suboptimal since they can't safely use delayed writes when ordering is required. The new approach outlined is called soft updating, and performs nearly as well as a memory-based file system. This scheme tries always to use delayed writes, allowing multiple writes to be buffered and coalesced. This, however, requires that information on the inter-operation dependencies be kept; since there may be multiple dependencies within a single disk block, dependency information is kept around for each metadata update, not just for each block in the disk buffer. This information makes it possible to write a block back to disk at any time, by undoing (rolling back) any updates within that block that still have pending dependencies. This makes the block written to disk consistent with respect to the current on-disk data. When the write is completed, any undone updates are re-established, and the in-memory block can once again be accessed. This paper and the next received Honorable Mentions from the Program Committee (see page 16).

The last file system talk was by David Kotz (Dartmouth College), who observed that even multiprocessor systems with large numbers of I/O processors often fail to provide the available hardware bandwidth to the application, which sits on top of the usual set of read/write/seek primitives organized around a coordinated file pointer. Parallel scientific applications are particularly problematic, since they tend towards complex strided access to discontiguous pieces of files, a very poor match for distributed buffer caches. Kotz's solution is to provide higher-level interfaces to the file system, passing the entire collective request down to the I/O processors, each of which can then optimize the handling of its portion of the operation. This scheme of "Disk-directed I/O for MIMD Multiprocessors" optimizes disk layout and scheduling, avoids problems of buffer management, and removes the guessing from prefetching and write-behind. This solution scales with the number of disks (modulo bus bandwidth) and is independent of the number of CPUs.

Tuesday's last set of talks centered on distributed shared memory (DSM). Robert J. Fowler (University of Copenhagen) presented "Message-Driven Relaxed Consistency in a Software DSM." They have built a software DSM machine, but with relaxed, "aggressively lazy" consistency. Memory state is made coherent only when messages containing explicit annotations are exchanged. This method combines data transfer and synchronization, and helps compensate for any mismatch between the hardware-imposed granule for access detection and the optimal size from the point of view of the abstractions of the application. Further benefit accrues from lazy evaluation, since only operations that explicitly require consistency incur the overhead to maintain it.

"Software Write Detection for DSM" was presented by Matthew J. Zekauskas (Carnegie Mellon University). Most implementations of DSM trap accesses and extend the virtual memory software to maintain consistency between competing application threads. The access faults used by these traditional methods are quite expensive, requiring many accesses to amortize the overhead. False sharing, when many objects fit into a single, large-grained page, also reduces efficiency. The approach taken here is to use the compiler and run-time support to detect and collect writes to shared data, and to trigger the consistency mechanisms. The benefits are keeping the operating system uninvolved in consistency operations, eliminating false sharing while directly supporting variable sized objects, and providing a detailed update history, thus minimizing the amount of data that needs to be transferred to maintain a consistent view of memory.

"The Design and Evaluation of a Shared Object System for Distributed Memory Machines," presented by Daniel J. Scales (Stanford University), outlined another software implementation of DSM built upon message passing. This method, like that of the previous talk, lets synchronization piggy-back on data messages; in addition, the user has to explicitly tag shared data as one of two types. Accumulators allow fine grained updates with built in mutual exclusion; single-assignment variables cause potentially many readers to block until the single writer has supplied the value, supporting producer/consumer relationships. This portable run-time system is claimed to improve communication efficiency in computations with complex access patterns, while providing a simple and easily-understood data sharing model.

## Wednesday, November 16

The first talk on Wednesday was "PATHFINDER: A Pattern-Based Packet Classifier," delivered by Michael A. Pagels (University of Arizona). Packet filters are code fragments that select packets of interest to a particular application, based on the contents of selected fields in the packet header. This project built a pattern-based mechanism which, given the pattern description, results in a decision tree able to classify incoming packets and tag them, i.e., to identify the path along which the packet will be passed. Patterns are described by tuples consisting of an offset (within the packet header), the length of the packet datum, a mask to be applied to the datum, and the value needed to produce a match. This mechanism was built in both software and hardware. The software version proved to be roughly twice as fast as the fastest (Mach) packet filter measured. The first FPGA hardware prototype, despite limited pattern store, can sustain 622Mbps speeds and can be expected to sustain gigabit speeds using existing technology.

Erich M. Nahum (University of Massachusetts) talked about "Performance Issues in Parallelized Network Protocols," using a parallel version of the x-kernel running in user space to examine the effects of choice of locking strategy on performance. His group found that a coarse-grained locking approach got better performance than finer-grained locking. Further, since packet ordering significantly impacts performance, where lock contention perturbed the ordering a simple FIFO queueing lock improved throughput substantially. They found that single-connection TCP parallelism was limited by the need to lock the connection state; while proposing the use of multiple connections to gain back the lost parallelism, they noted that this implied punting the responsibility for managing ordering among connections up to the application. Support for atomic increment (in lieu of lock-increment-unlock sequences) was shown to gain 10-20%. Finally, they got some speedup from keeping a small cache of processor-local data structures that are frequently handled by the x-kernel.

Ronald C. Unrau (University of Toronto) described his group's "Experiences with Locking in a NUMA Multiprocessor Operating System Kernel." They have arrived at a hybrid locking scheme, using both fine and coarse grained locking in an attempt to get the highly concurrent behavior of the first with the low latency and space overhead of the latter. This approach uses a coarse lock to protect several data structures, which can only be held for short periods of time. Finer grained locks protect individual structures, and are held for longer periods, but taken under the coarse lock. This approach complicates the locking protocols, but reduces second order effects such as memory and bus contention, while exponential backoff mitigates so-called thundering herd effects. A technique called hierarchical clustering replicates data that is primarily read-shared to bound the contention on shared structures by constraining the number of CPUs that can attempt access.

Chao-Hsien Lee (National Chiao Tung University) presented "HiPEC: High Performance External Virtual Memory Caching," talking about the work his group has done to allow each application customized control over the page replacement policy underlying its virtual memory support, in order to reduce thrashing. An application programs its own custom policy in the HiPEC command set, keeping the resulting table in user space. When a page fault occurs, the kernel interprets the application's policy and resolves the fault in accordance with that policy. Applications using this mechanism also have a private page frame list, to avoid interference between concurrent applications using differing policies. The actual policy information is interpreted from within the kernel to avoid the added domain crossing overhead which would have resulted from passing control out to the application on each fault. Their results showed little added overhead and the possibility to significantly improve performance for memory-intensive applications with well-understood access patterns.

Pei Cao (Princeton University) spoke about "Implementation and Performance of Application-Controlled File Caching," which, like the previous talk, allowed user-space applications to gain control over the replacement policy used in maintaining their (process-private) file buffer cache. Their approach applies the notion of working set-to-file caching. A two-level mechanism first associates a priority each file, and then associates a policy with each priority. The kernel allocates cache blocks to processes; within a process, the lowest priority set of blocks is chosen for replacement, and the policy chooses blocks within that set. There are also mechanisms to temporarily change the priority of a range of blocks within a file, and to ensure that foolish or malicious processes can not hurt other processes. As an example, this scheme allows a database to set its index files to be a higher priority than the data files, keeping them around for future re-use, while selecting whatever policy is appropriate for each set.

David R. Cheriton (Stanford) presented "A Caching Model of Operating System Kernel Functionality," which is the supervisor-mode component of the V++ operating system. This effort seems to be another attempt to put the "micro" back in micro-kernel. The approach taken is to build a minimal layer which caches threads and address spaces the way that hardware caches the contents of memory. User-mode application kernels (read: servers) handle the loading and storing of these objects, and implement application-specific management policies and mechanisms (read: OS personalities). As with previous micro-kernel projects, the result is promised to be faster, smaller, more scalable, modular and robust than all prior efforts. Good use is made of "address-valued signals", similar to UNIX signals but with a pointer-sized value, which provides asynchronous messaging built on top of shared memory.

The last slice of Wednesday afternoon was saved for a panel discussion, whose topic was "Radical OS Structures for Extensibility." Five panelists spent a few minutes advocating their own approach to building operating systems. Brian Bershad (University of Washington) talked about Spin, Larry Peterson (University of Arizona) touted Scout, and Frans Kaashoek (MIT) peddled Aegis. David Cheriton advocated the caching kernel approach of V++, while Steve Lucco (CMU) plugged software fault isolation. The jokes were first-rate, and it seemed that the panelists put as much effort into bashing the competition as advocating their own work. The panel was quite enjoyable, and if DOS finally comes to rule the world, any of these guys could find work in standup comedy, at least for audiences of OS developers. Unfortunately, the technical information content was practically nil, with the usual set of buzzwords (smaller, faster, more robust!) used to justify all of the efforts, and kernels shirking ever more responsibility off to applications. The conference proceedings do, however, contain single-page summaries from each panelist, with enough content to be worth reading.

## Thursday, November 17

Thursday began with another set of papers on distributed shared memory. David K. Lowenthal (University of Arizona) talked about "Distributed Filaments: Efficient Fine-Grain Parallelism on a Cluster of Workstations." Filaments are a light-weight abstraction, conceptually just a stateless thread that is given some arguments and a function to execute. Filaments come in three flavors, each appropriate for a different sort of computation. Run-to-completion filaments execute once and then terminate, useful in applications such as matrix multiplication. Iterative filaments execute repeatedly, performing a barrier synchronization after each execution of all the filaments; a Jacobi iteration could use these. Fork/join filaments recursively fork off new filaments and wait for them to complete, useful in divide-and-conquer solutions such as adaptive quadrature. Filaments have no private stack (they are executed by a server thread that does have one) and are blocked along with the underlying server thread during fault processing. Multiple server threads are created per processor, so that another filament may be run while the block thread awaits the page. This package is portable, and implements multi-threaded DSM on a variety of hardware, including clusters of workstations and at least one multicomputer.

"Integrating Coherency and Recovery in Distributed Systems," presented by Michael J. Feeley (University of Washington), is a new twist on the usual DSM, meant to support collaborative design efforts rather than parallel programming. The result is a log-based coherency mechanism layered over a persistent store, where the logs, as in the database world, provide both recoverability and coherency. This method separates the granularity of synchronization from that of updates and invalidations, performing particularly well when the size of update is smaller than the underlying virtual memory page. The logs might be viewed as a collection of distributed caches, implementing DSM for the applications.

Paulo Ferreira (INRIA and Université Pierre et Marie Curie), presented Garbage Collection and DSM Consistency, a design which minimizes cross-node communication to provide efficient garbage collection over DSM. This project is based on the observation that, in a weakly consistent DSM system, the memory consistency requirements of the garbage collector are less strict than those of the applications. Thus the garbage collector reclaims objects independently of other copies of the same objects, without interfering with the DSM consistency protocol. Distributed cycles of dead objects are scavenged, and reliable communication support is not assumed.

The final technical session had three talks on memory management. The first, "Software Prefetching and Caching for Translation Lookaside Buffers," was by Kavita Bala (MIT). The number and cost of software TLB refills was reduced by keeping a cache of TLB entries, and by prefetching the entries needed by a process involved in a messaging RPC: those for the PC, stack pointer, and message buffer(s). The cache of TLB entries provided a hot path that handled over 90% of all TLB misses, and the prefetching prevented multiple traps resulting from a single attempted system call. Cascading TLB misses were also eliminated, though this result is less general, since it's an artifact of chips which both handle TLB misses in software and use virtually-mapped page tables.

"Dynamic Page Mapping Policies for Cache Conflict Resolution on Standard Hardware," presented by Theodore H. Romer (University of Washington), tackles the problem of cache conflict misses, both with hardware and with a software approximation of that hardware. On hardware with large, physically-indexed direct-mapped caches, a conflict miss occurs when two memory locations compete for the same cache line, even though the cache may be large enough to hold the current working set. Most approaches to solving cache conflicts involve static mapping policies, which assign a page frame to a virtual page at page-in time, on the basis of spatial locality (page coloring) or temporal locality (bin hopping). The problem inherent in these approaches is that access patterns may vary over time, making any static solution sub-optimal. A variety of mapping policies were simulated, showing that a dynamic mapping policy that adjusts to the ongoing behavior of the program outperforms any static policy for some workloads. Further, the best performance can be had by a simple hardware enhancement called the Cache Miss Lookaside Buffer, which tallies the misses of various physical page frames, allowing pages mapping to the same cache line to be moved if the number of misses becomes excessive.

"Cooperative Caching: Using Remote Client Memory to Improve File System Performance," presented by Michael D. Dahlin (University of California, Berkeley), examined the benefits of workstations and shared servers working together to provide a larger effective file cache. The simulations, based on traces from various workloads, showed that relatively simple cooperative algorithms can halve the number of disk accesses and improve read response time by over 70% in typical workstation cluster configurations. These results become even stronger as we follow current trends toward faster networks and CPUs (relative to disks).

The final afternoon session was devoted to a Mach/Chorus workshop, where a half dozen informal presentations were made. Papers corresponding to these are not included in the conference proceedings, but hard copies (and pointers to online copies) were provided by the authors.

Frederic Ruget (Chorus Systèmes and Université Joseph Fourier) talked about Micro-Kernel Support for Trace-Replay, in which Chorus was instrumented to support tracing of a variety of events, including system calls, traps and asynchronous interrupts. This is intended to provide support for statistics gathering, profiling and tuning, visualization and understanding of system behaviors, as well as allowing for testing and fault injection.

Dejan Milojicic (OSF Research Institute) spoke about his work on "Concurrent Remote Task Creation in Mach." He found a variety of problems that caused remote task creation to scale up poorly, ranging from slow remote IPC, through pointless paging directed at a single node, to inefficient VM operations resulting in ever-increasing shadow chain length. Copy-on-write, a mainstay of Mach VM, is sub-optimal except in the local case; copy-on-reference results in much less network traffic in distributed task creation. Further work will use a tree-structured mechanism to reduce the bottleneck of sequential creation requests when large numbers of nodes are involved.

Michael Condict (OSF Research Institute) spoke about the success of his project in attaining "Microkernel Modularity with Integrated Kernel Performance." The thrust of this work was to load servers into supervisor mode once they're sufficiently stable to be trusted, and to make RPC take advantage of this "collocation" by automatically using the most efficient mechanisms permitted by location of sender and receiver, since crossing protection domains is expensive. A variety of lesser performance problems were also corrected, to produce a micro-kernel system that performs on a par with its monolithic equivalent.

"Operating System Support for Coexistence of Real-Time and Conventional Scheduling" was presented by David B. Golub (Carnegie Mellon). The usual approach to this problem is to have many priorities, but divided into segments, with one set treated specially to provide some semblance of real-time response. He noted that both rate-monotonic and earliest-deadline first policies have some problems, so more flexibility is needed. His approach is to associate with each priority a scheduling policy, and to provide per-policy scheduling parameters. When a scheduling decision is to be made, all threads in a higher priority beat those in a lower priority, and the policy selects threads within the priority, using the scheduling parameters. The processor set and priority interfaces were extended to accommodate these changes.

"A Memory Management Mechanism and An External Resource Manager Interface for Continuous Media Objects" was presented by Satoshi Moriai (Nippon Telephone and Telegraph Corporation). This project addressed the needs of distributed multimedia to allow dynamic control of the quality of service guaranteed to an application handling continuous multimedia requests. The focus was on supplying predictable VM performance for two sorts of real-time behaviors. In temporal paging, the active region of VM migrates over time, while in real-time projection, the contents of a fixed region change continually over time. Several techniques were discussed, which included wiring memory, sharing memory between devices and applications, as well as a virtual ring-buffer with timestamp-based pageout and the use of page swapping to guarantee page availability as old pages are discarded.

Clifford W. Mercer (Carnegie Mellon) spoke "On Predictable Operating System Protocol Processing." He talked about the need for timely resource management, and the need to distinguish time-constrained packets. The general approach taken seemed to be resource reservation, including network bandwidth, CPU time for both the protocol and the application, as well as reserving the "attention" of the server (which shouldn't be single-threaded).

## Best Paper

The best paper at OSDI was *Lottery Scheduling: Flexible Proportional-Share Resource Management* by Carl A. Waldspurger and William E. Weihl, MIT.

Papers winning "Honorable Mention" were *Metadata Update Performance in File Systems* by Gregory R. Ganger and Yale N. Patt (University of Michigan); and *Disk-directed I/O for MIMD Multiprocessors* by David Kotz (Dartmouth College).

# ROSE '94 – The Romanian Open Systems Conference & Exhibition Report

*by Alexandru Rotaru*
*<arot@guru.ro>*

On November 3-5, 1994, GURU, the Romanian UNIX User Group, hosted the EurOpen Governing Board Meeting after their International Open Systems Event – ROSE '94 (Conference & Exhibition) in Bucharest.

Some of the important speakers invited by GURU were: Richard Stallman, Elizabeth Zwicky, Philip Zimmermann and Chet Ramey. Unfortunately, Ms. Zwicky was ill and could not attend. Some other speakers provided on behalf of EurOpen were: Kim Biel Nielsen, the chairman of EurOpen, Simon Kenyon (IUUG), Mick Farmer (UKUUG), Etienne Remillon (AFUU), Jean-Michel Cornu (AFUU), Marten van Gelderen (NLUUG), Emile van Dantzig (NLUUG), Sergei Kuznetsov (SUUG), Mario Zagar (HrOpen), and Zoltan Porkolab (HUUG).

A big attraction for all participants was the spontaneous discussions outside the conference hall after the presentations. For example, after Richard Stallman's 20-minute presentation the outside discussion lasted almost an hour.

Twenty-six companies from Romania and abroad joined the Exhibition Area. This year, most of them provided speakers from the technical departments from Romania and abroad. Their presentations were not strictly promotional, but technical too. Full Internet access was possible for all the visitors. Some demonstrations with different navigation tools were organized by the companies. In the GURU's booth free software was distributed and practiced. A lot of Proceedings, magazines and advertising materials from EurOpen, USENIX, UniForum, X/Open and GURU were displayed in the same booth.

Two Romanian Computer publication groups and some newspapers were present. Up to now they continue to publish interviews from the meetings.

There were more than 700 subscribed participants in the conference and more than 1500 visitors during the events. This number is very large for a Romanian conference especially on Open Systems, a subject which is not very well known in Romania.

GURU had some local technical support for organizing ROSE '94. All the organizers used pagers from a sponsor paging company and simultaneous translation was provided by another sponsor.

Thanks to all who helped and made the conference a success.

# Community News

**Rich** *<rsalz@osf.org>* and **Martha Salz** are the proud parents of Carter James, born August 5 (9 lbs 1 oz., 22"). Older brother Braedon (22 months at the time) is learning to adjust.

**Susan Richter** *<richter@rand.org>* and **Win Bent** *<whb@usc.edu>* are pleased to announce their marriage on October 22, 1994. The couple met at the evening reception of the Summer '92 USENIX Conference in romantic San Antonio (they were the only ones actually paying any attention to the band!). Their email exchange over the next several months significantly impacted Internet traffic loads, until Win moved out to sunny Southern California from Bell Labs in New Jersey.

We regret to announce the death of **Jean Serge Banino**, one of the authors of the *DUNE_iX* paper published in *Computing Systems* 6.4 (1993), in an automobile accident.

# 1995 Lifetime Achievement Award Announced

The USENIX Lifetime Achievement Award recognizes and celebrates singular contributions to the UNIX community in both intellectual achievement and service that are not recognized in any other forum.

At the Technical Conference in New Orleans, the third annual Lifetime Achievement Award was presented to Tom Truscott, Steve Bellovin, and Jim Ellis for their work in creating USENET, announced exactly 15 years ago at the USENIX Conference in Boulder, Colorado. The award also honors the thousands of participants and supporters who have contributed to USENET over the years, and who are far too numerous to name.

The recipients of this award, known as "Keepers of the Flame," receive an original glass sculpture entitled "The Flame."

The award was first presented in 1993 to The Computer Systems Research Group, honoring 180 individuals for their contribution to the CSRG effort. In 1994, the award was given to Michael Lesk for inventing UUCP, and Van Jacobsen for his work on making TCP "Industrial Strength."

The award honors those who have materially changed the world with their contributions to network technology.

# Web News: Online Abstracts and Full Papers

*by Carolyn S. Carr*
*Publications Manager and "Webster"*

Have you checked out our Home Page on the Web? The USENIX Resource Center *<http://www.usenix.org>* maintains the most current listing of upcoming events, membership and order information, sample articles from past publications, and access to the USENIX Online Library and Index.

Last fall we linked the abstracts for all papers from the 1994 USENIX proceedings to our Web site. Within the next month we will make the full text (ASCII and PostScript) of each paper available to all members.

Watch your postal mailbox. If you are a paid up member of USENIX, you will receive a new membership card with a password in the next month. This password will provide access to the refereed papers published in the 1994 proceedings on our Web server. As future proceedings are published, we will link the abstracts and full papers.

If you are an author whose paper is accepted for an upcoming conference or symposia, keep in mind that the ASCII version that you submit will be published on the Web. Please take care in sending us a clean, un-coded, readable version.

SAGE, the System Administrators Guild, is dedicated to the advancement and recognition of system administration as a profession. In just two years, SAGE's membership has increased steadily, and there is growing recognition of SAGE as a representative in system administration issues. SAGE brings together system and network administrators for:

- professional and technical development,
- sharing of problems and solutions,
- communicating with users, management, and vendors on system administration topics.

SAGE News Editor
- Bryan McDonald *<bigmac@usenix.org>*

SAGE Board of Directors
- Elizabeth Zwicky, President *<zwicky@usenix.org>*
- Paul Evans, Secretary *<ple@usenix.org>*
- Peg Schafer, Treasurer *<peg@usenix.org>*
- Paul Moriarty *<pmm@usenix.org>*
- Pat Parseghian *<pep@usenix.org>*
- Steve Simmons *<scs@usenix.org>*
- Pat Wilson *<paw@usenix.org>*

SAGE Working Groups

| Group | Chair |
|---|---|
| *sage-certify* | Arch Mott |
| *sage-edu* | Ron Hall |
| *sage-ethics* | Ed Gould |
| *sage-jobs* | Tina Darmohray |
| *sage-locals* | Rene Gobeyn |
| *sage-online* | Mark Verber |
| *sage-policies* | Lee Damon |

You can contact these groups via email at *<their-name@usenix.org>* for example, *<sage-certify@usenix.org>*.

SAGE Discussion Groups
*sage-security*
*sage-managers*
*sage-outreach*
*sage-pt*

SAGE Online Services
Email server:
*majordomo@usenix.org*

FTP server:
*ftp.sage.usenix.org*

WWW URL:
*http://www.sage.usenix.org*

SAGE Supporting Member
Enterprise Systems Management Corp.

SAGE
THE SYSTEM ADMINISTRATORS GUILD
NEWS

# SAGE Elections Results

The results of the elections for three director seats on the SAGE Board of Directors for the 1995-97 term are as follows:

*Elected for 1995 & 1996, two-year term*

| | |
|---|---|
| Kim Carney | 359 |
| Bryan MacDonald | 298 |
| Hal Miller | 262 |
| *Not Elected:* | |
| Adam Moskowitz | 171 |
| Jim Duncan | 155 |
| Total Ballots Cast | 467 |

Four directors (Paul Evans, Paul Moriarty, Pat Wilson, and Elizabeth Zwicky) will return to the SAGE Board of Directors for 1995. These Board members will complete their two-year term at the end of 1995.

# LISA VIII Conference Reports*

## September 19-23, 1994

### A Practical Introduction to SNMPv1

*Phil Draughon*
*Summarized by Andy Poling*
*<andy@jhunix.hcf.jhu.edu>*

In this invited talk, Phil described the Simple Network Management Protocol (SNMP) and its potential uses. He stressed the fact that it should really be SMP since it is not limited to network management.

Phil started by looking at why we want to monitor equipment/resources, and then described some of the ad hoc methods used in the past to monitor, comparing them to the use of SNMP. He also mentioned the possibility of "proxy man-

* Other reports from LISA VIII can be found in the December 1994 *;login:*, pp. 5*ff.*

agement" of non-SNMP capable devices via an SNMP-capable machine.

The most important (and therefore lengthy) part of Phil's talk was his explanation of the Management Information Base (MIB). He explained that the MIB describes a hierarchical structuring and type declaration for arbitrary data. He then went over some examples to illustrate. His overhead slides are available as *<http://www.sage.usenix.org/lisa/lisa8/draughon-it-snmp.ps>*.

## T06: Performance Monitoring & Tuning

*by Mark Staveley, consultant*
*Summarized by Larry J. Miller*
*<ljm@halsp.hitachi.com>*

Is your system running like an old Model T, when you would like it to perform like an Alfa Romeo? Then this was the LISA VIII Invited Talk for you! Mark gave an excellent presentation on what to monitor, what the results of all that data means, and what to do to correct deficiencies in your system performance. He stated that "All system performance issues are basically resource contention issues." So the idea is to find WHICH subsystem is really in trouble – and only then make changes.

The major subsystem breakdown Mark used was: KERNEL (system tables are over 90% of the problems), MEMORY (buy more if possible – but there are tricks to get more back), DISK I/O (bigger disks hurt performance; one disk per controller for best performance), NETWORK (here is another major contributor to slow performance – can be faulty hardware, or too high a load per segment), and NFS (check the retransmission rate). Armed with publicly available programs such as "pstat," "vmstat," "iostat," "sar," "netstat." and "nfsstat" you can monitor your system, gather loads of data, and determine which subsystem is slowing performance.

Mark gave excellent examples of how to specifically monitor each subsystem, how to interpolate the data, and then what to do to increase performance. Mark stated his firm belief in "MONITOR, MONITOR, MONITOR." Do it every day, graph the data, and gather statistics over time and you too could have an Alfa Romeo system!

## Automation, the Sequel

*Summarized by Steve Wright*
*<wright@CS.Scarolina.EDU>*

### SENDS: a Tool for Managing Domain Naming and Electronic Mail in a Large Organization

*Jerry Scharf, Sony Electronics and Paul Vixie, Vixie Enterprises*

In a large organization it is difficult to manage DNS and mail configuration information. This paper documents a set of simple tools that provide management of host and electronic mail information.

It provides central management and allows information that is determined by constituent groups to be controlled remotely, collected and processed centrally and distributed to the whole user community.

The authors code was to be available for ftp on *ftp.vix.com* in */pub/vixie/SENDS/sends.tar.qz.*

### Getting More Work Out Of Work Tracking Systems

*Elizabeth D. Zwicky, Silicon Graphics*

This software was written to automate problem detection. The motivation was to have software detect and "complain" about problems before users did. There are a lot of dangers inherent in having "fixes" automated. The decision was made to have a human notified to make corrections that the software found.

The project was implemented at SRI as a set of small programs which provided flexibility and were easily expanded to provide more features over time. The software was reimplemented upon moving to SGI as a single monolithic program, eliminating a lot of redundancies found in the original programs.

### Managing the Ever-Growing To Do List

*Rémy Evard, Northeastern University*

A common and growing problem is keeping track of tasks. This paper discusses a software system that was developed to stay on top of the growing list of things to do. In addition the paper addresses the question of mechanisms for using the software to provide better support for a user community.

The system is centered around a mailing list to system administrators that is monitored by a staff member on the "hotseat." Information is sent to individual and group "to do lists." Requests can be accessed and their status determined by users while they are in process. As solutions are produced users are informed by mail and FAQ documents.

Available by ftp from *ftp.ccs.neu.edu* as */pub/sysadmin/req.tkreq.*

## Peek a Boo – I Can See You

*Summarized by Steve Wright <wright@CS.Scarolina.EDU>*

### Monitoring Usage of Workstations with a Relational Database

*Jon Finke, Rensselaer Polytechnic Institute*

This paper presents a mechanism for collecting large amounts of information from a large number of workstations, standardize format, and storing it in an Oracle database. Once collected, the tools available as part of the database allow the extraction of many detailed reports on users and the use of facilities. The graphical capabilities produce output that make it easy to spot trends in the demographic information.

Information on the availability of Simon is available from *ftp.rpi.edu* in */pub/itsrelease/Simon.Info* or you may browse it at *<http://www.rpi.edu/~finkej/Simon.html>*.

### Adventures in the Evolution of a High-Bandwidth Network for Central Servers

*Karl L. Swartz, Les Cottrell, and Marty Dart, Stanford Linear Accelerator Center*

This paper studies the evolution of the server networks at SLAC. Difficulties encountered in the deployment of FDDI are documented. They found that FDDI was not the best solution for their server networks. The combination of cost, interoperability problems and the difficulty monitoring an FDDI network made the implementation of a network using ethernet switches a better choice for their environment.

The use of Ethernet switches increased performance over an unswitched network. It has the advantage of using the more mature Ethernet technology and existing or relatively inexpensive interfaces.

### Pong: A Flexible Network Services Monitoring System

*Helen E. Harrison, Mike C. Mitchell, and Michael E. Shaddock, SAS Institute, Inc.*

Pong is a highly configurable monitoring tool which "pings" individual services at predefined intervals and executes appropriate actions. Ping queries can return misleading information as machines which are operating minimally will report that they are "up," while refusing other services.

Pong uses three protocols to check the viability of a machine. This provides much more useful information than ping alone.

There are two utilities that were created to provide an interface to pong information. Pongstat connects to pong and collect summary information. King_pong is a GUI interface to the pong information.

For availability information contact *<heh@unx.sas.com>*.

## Software Configuration
## The User Environment

*Summarized by Paul Anderson <paul@dcs.ed.ac.uk>*

These two sessions included papers on three main topics; organizing and distributing software packages from a central repository, configuring and customizing the user environment, and making UNIX home directories available to PC users.

John Roulliard presented a variation of the familiar Depot system which is being used for managing software packages at the University of Boston. In "Depot-lite," the filesystem layout has been changed to group together all files for a particular platform; this allows the system to be adopted more easily by small clusters which do not use an automounter. Software installation is also simpler and some support is provided for multiple simultaneous versions.

Thomas Eirich described a tool called "Beam" for updating local copies of software packages from a master server. This program updates by "pulling," using NFS to access the master copies of the files. Beam is written in Perl and is highly configurable, but it appeared to add little to previously reported systems such as "lfu" and "Nightly."

A paper by Rémy Evard and Robert Leslie from NorthEastern University won the award for best student paper. This describes a higher level configuration language for specifying a user's preferred environment. A program called "Soft" generates startup scripts, for different shells, from the same configuration specification, and these scripts can be cached to improve shell startup times. Changes in the details of a package installation do not normally require changes to the user configuration files and new startup scripts are generated automatically.

Carl Hauser from Xerox described their system for configuring the user environment. This is based on a mechanism similar to Furlani's Modules, but the resulting environment is cached to avoid the delays associated with enabling lots of packages. A background process is forked to recompute the environment cache so that logins are always fast at the expense of a slight delay in changes to the configuration.

The "BNR Standard Login" described by Christopher Rath is a yet another scheme to provide a method for users to configure their environment without the maintenance problems associated with complicated "dot" files. This system has no cache but uses a dedicated C program to interpret the configuration, and provides persistent environment variables for applications.

The final paper of these sessions described some modifications to the PC-NFS daemon which is used to export UNIX home directories to large numbers of PCs across the campus at the University of Kent. The lack of a PC automounter is overcome, and security is improved, by having the pcnfsd locate the appropriate server and authorize it to export just a single home directory to the appropriate PC.

## Response to LISA Summary on GASH

*by Jonathan Abbey*
*<jonabbey@arlut.utexas.edu>*

Bruce, I appreciate your summary of my presentation on GASH at LISA VIII in the December 1994 *;login:*, but I'm a little disappointed that the summary doesn't really adequately explain just what it is that GASH is, and that my presentation wasn't clear enough to prevent confusion.

GASH is fundamentally a sophisticated, easy-to-use, highly automated NIS and DNS management system. GASH provides mechanisms for sharing and delegating authority over subsets of the NIS and DNS databases. In addition, GASH was designed around a set of conventions that make system administration tractable for a network on a 1,000 machine scale.

The future projects and account validation stuff is quite a bit less important than the basic character of GASH, and in fact the account validation stuff is very specialized to our environment here.

The MIT/Athena program referred to be a member of the DEC Athena project at LISA VIII was Moira.

We have been working steadily on producing version 1.03 of GASH, and will be announcing its release through our mailing lists and various prominent newsgroups soon. We hope that our efforts will be of benefit to as wide a range of UNIX administrators and UNIX shops as possible, and would appreciate an addendum to the summary given, as appropriate.

# User Interface Development Tools: Remarkable Promises, Occasional Delivery

*by Cindy A. Harris*
*<cah@lonewolf.com>*

If you are involved in creating and implementing complex user interfaces, you have almost certainly noticed the proliferation of ads in industry publications for tools that claim to enable the most novice of developers to create the most sophisticated of user interfaces with the click and drag of a mouse. Not only can you create beautiful interfaces, you can also transport them in the wink of an eye to a multitude of platforms and interface them to a variety of databases.

As usual in our industry, it turns out that the claims, while true at some level, fall short in a number of areas. Having spent the last year surveying the field, I thought to share a number of observations that you may find useful when you decide to take the plunge and select a tool that may end your recurring nightmares about user interface development in X, Motif, Windows, or your favorite flavor toolkit.

## GUI Builders, toolkit abstractions, and other buzzwords

User interface development tools tend to fall into three basic categories: GUI builders, toolkit abstractions, and user interface management systems. There is also a fourth group that I would call "database oriented tools." While they often look alike in ads, their philosophies and capabilities are very diverse, and understanding the differences between them will allow you to choose a tool from a category that caters to your needs.

GUI Builders can be very useful and cost effective for small projects intended as prototypes or single-user systems. They allow the user to create a static interface, usually in a WYSIWYG fashion, and to generate a template that contains prototype calls to, and/or callbacks from, your chosen graphics toolkit. The programmer is left with the task of understanding the toolkit well enough that she can fill in the blanks to create any behaviors (a button reacts when clicked), and to link user actions to application tasks (clicking on a button brings up a customer record).

These tools work best when used to outline the picture of the screen and generate the basic template and then ignored for the rest of the development process. Most have limited ability to reverse the process and read generated code back in (especially once it's been modified) to enable the tool to be used for subsequent iterations of the design process. Use of these tools results in a system in which the application must be intimately aware of the user interface, and vice versa. Changes to either usually require changes to both, and the tool generally can't be used to accelerate the process. Transportation to a different platform generally means re-generating the basic template on the new platform and customizing the application to work with the new toolkit (which implies that the programmer is familiar with the new toolkit). Of course, these tools are usually priced at under $500, which makes them tempting if you're developing for a single toolkit that you already know, and want to speed up the process a bit at the front end.

Toolkit abstractions offer some specific advantages over builders, but still provide some challenges. The "abstraction" is a generic toolkit that provides an interface layer on top of underlying technologies. The programmer uses generic "widgets" to describe a user interface, and the toolkit abstraction makes calls to the "real" toolkit (e.g., X11, MS-Windows, Macintosh). This makes applications developed much more easily transportable to other platforms, since the programmer uses only one generic toolkit to implement applications that will run on multiple platforms.

But there is a catch: the abstraction is limited to describing only elements that are present in every supported toolkit. This is known as the "greatest common subset" (often referred to as the "least common denominator"). If you want to use a widget that is only available on some of the toolkits, you're out of luck. (A few tools get around this by providing their own set of custom widgets that includes a superset of widgets from the toolkits they support.) Most toolkit abstraction packages also include a GUI builder, which gives them the same benefits and problems as standalone builders. If you need to quickly create an application that will run on multiple platforms, don't anticipate many changes to it in the future, and are willing to work with a subset of the widgets available in your toolkits, a toolkit abstraction may be what you need. Prices range from $500 to about $12,000 per development license. Some also require runtime licenses.

User interface management systems are the newest and smallest category of tools on the market today. I am aware of only three commercially available UIMS's that come close to meeting the accepted technical definitions of a UIMS (the Seeheim and Arch models). The tantalizing promise of a UIMS is that it can provide all the benefits of a toolkit abstraction without their "greatest common subset" restriction and also allow the application "engine" to be separated from the user interface specification (dialogue) so completely that changes to either one or to the toolkit itself (e.g., upgrades) do not affect the others in any way. At worst, a UIMS should allow you to make a change to any of the components, recompile only the affected component, and re-execute the changed components. Ideally, you should be able to change and re-compile any one component "on the fly" without having to touch any of the others.

These tools vary in their approach to creating actual code. Most work like traditional builders, allowing the programmer to specify the interface in an abstraction language, then generating code specific to the target toolkit and allowing links to application code. Some encourage you to compile and link the user interface (written in abstraction language) and the application (written in your favorite language) directly, skipping the code generation step, thus avoiding the temptation to meddle with the generated code and render the tool useless after the first iteration. Some tools take this concept one step further by only generating executables. In this scenario, the link between the application and the user interface is made via an IPC mechanism, allowing the application to be distributed across multiple platforms.

User interface management systems have other less obvious advantages, owing to their separation of application intelligence from user interface. For example, applications are easier and quicker to debug, since application functionality is localized and separated from dialogue functionality. Changes to the user interface can be made much later in the application development cycle than with tools that require generation of code, allowing more time for the end user to review and modify the dialogue, and resulting in an interface that doesn't stop at "good enough." Maintenance and upgrades are quick and simple, as they tend to affect only a single module. And ports to additional platforms are a breeze, since machine specific parts are isolated in separate modules (these modules may even be left of the specific machine in the distributed systems).

With all of these advantages becoming more widely known, more and more tools are beginning to claim membership in the UIMS club. But beware: many do not even come close to the required ability to separate application from dialogue, and are really more toolkit abstraction than UIMS. By and large, these tools range in price from $5,000 to $12,000 per development platform, and may require runtime licenses as well.

Database-oriented tools are a fourth category whose membership crosses the other three. What these have in common is a superior ability to implement database-retrieval-oriented interfaces and forms. They are marketed by all of the major database vendors as adjuncts to their database tools, and can be a good bet if you are developing transaction-oriented distributed database applications. Don't plan to use them for other kinds of applications, however, their capabilities are generally more limited or more difficult to use than those of non-database-oriented tools under these circumstances. Of course, the opposite can be said of the other types of tools if you're developing database applications. Prices range from $5000 to $25,000 per development platform, and almost all require a runtime license.

## The myth of drag-and-drop productivity

If you've never tried a drag-and-drop dialogue editor, you owe it to yourself to get your hands on one and find out what it's about. But don't buy a tool because you love its editor. I have found that visual editors are great for the initial layout of the static portion of a user interface. But if you want to create complex dynamic user interfaces, or if you plan to be making a lot of changes to your interface once you have generated the code, don't plan on getting a lot of productivity from these tools. As previously mentioned, once code has been generated or an application integrated, regeneration capabilities are limited, and the editors are not useful. And implementation of dynamic behaviors (i.e., buttons and labels appearing or disappearing depending on particular conditions) almost always requires programming and use of either the native toolkit (in the case of GUI builders) or the tool's API (in the case of toolkit abstractions and UIMS's) rather than use of the editor. So don't factor in use of an editor when calculating how these tools will improve your productivity if your application will have ongoing changes or has a significant dynamic component: they WILL improve it, but only in the early stages of the implementation process.

## Taking dynamics into account

The last observation I will make is that appearances (and even demos) can be extremely deceiving. Most vendors will demonstrate creation of a dialogue "from scratch." But look closely: that dialogue probably does not have any dynamic elements at all. If you need to create dynamic interfaces (and now that the capability is available, it's obviously necessary to use it), breeze right past the demo of the dialogue editor and take a good, close look at the API. That's what you're going to be using to implement your dynamic behaviors (unless you're using a builder that doesn't have an API, in which case you'll use the native toolkit). This can be an excruciating process if you are developing a large system, or one that will have a large number of changes (which almost always happens), so be sure you clearly understand what you might be getting into. Some API's are a large, complex language in themselves, while others are elegantly simple and easy to use.

## The bottom line

Once you decide to take the plunge, try to decide what type of tool best meets your needs first, then define an evaluation process. This will enable you to tailor the process so that it focuses on differences between tools that have similar world views, and avoid cross-type comparisons. It makes little sense to give a low score to a UIMS because it has no associated screen layout tool if it allows you to implement the kind of dialogues with dynamic behaviors that you need to create four or five times faster than with a really impressive GUI builder (or a UIMS with an associated builder).

Here are a few critical things to include for each tool type. If you plan to create simple, largely static user interfaces, you'll be evaluating GUI builders, and should be concerned about how easy it is to draw your interface. Ask also how easy it is to write callback code and add widgets, and how good a job the tool does at reflecting changes to the callback code.

For toolkit abstractions, evaluate how accurately the abstraction references the underlying toolkits that you need to use, and how well the finished product adheres to the native "look and feel." If the abstraction does not contain all of the widgets that you think you need, ask whether it is possible or desirable for a user to extend it, and, if so, how hard that is (some have extended libraries already available for purchase).

Evaluating a UIMS is somewhat of a different proposition. Here, you shouldn't need to worry about how easy it is to do callbacks – UIMS's use a different (and much simpler) mechanism to achieve the same effect. Instead evaluate how completely components are separated (a change to the dialogue, for example, should not entail a re-compilation of the application or technology components). If you are evaluating UIMS's, you probably plan to create complex dynamic user interfaces, so you should also take a close look at how easy it is to implement those behaviors (this is also a good thing to look at for toolkit abstractions). And if you care about things like re-usability of code, you will probably want to evaluate the object-oriented characteristics of any of the three types.

It is clear from the number of tools on the market today that there is a crying need for tools that will make it easier to design, specify, implement, port, and maintain user interfaces. Many of those tools can address these needs to some degree for developers in particular circumstances. But as end users continue to become more sophisticated in their requirements and the various interfaces technologies become richer in function, most programmers still find themselves resorting to less than optimal tools and lots of coding. So, go into the evaluation process for these tools with an open mind, but don't hesitate to conclude the process by choosing not to make a purchase if the tools truly don't fit your circumstances. There's nothing more frustrating than spending good money on a tool that sits on the shelf after the first 30 days of use!

*Cindy A. Harris is the Manager of Marketing for LoneWolf Systems, Inc., a company dedicated to helping developers create superior user interfaces through consulting and systems integration services, and by providing tools like the Alpha User Interface Management System.*

# Some Thoughts on the Economics of Information

*by Scott Hazen Mueller*
*<scott@zorch.sf-bay.org>*

Many folks have been saying a lot of things about the information that will be available in the future. Large telecommunications companies, among others, are planning to make a lot of money off of easily accessible information. Other people, steeped in the free information/free software culture, are up in arms about for-pay information services. I have a few thoughts of my own, sure to please neither side.

First off, the information service business has two main components. There is the transport side, which gets the information from the producer to the consumer. The Internet, X.25 networks, and the PSTN (Public Switched Telephone Network) are examples of transports. The other component is the content side. Producers of content range from media conglomerates, such as Paramount, all the way down to the folks who hand out pamphlets on street corners.

Industry news magazines make it abundantly clear that the large companies have their own vision of the future. In that vision, the transport companies have connected with the content producers, and giant conglomerates will produce and deliver entertainment, and possibly some education, on a pay-per-view basis. These conglomerates want to sell video-on-demand, music-on-demand, encylopediae by the page (or fraction of a page), you name it. In this vision you'll pay for data every time you use it.

Furthermore, none of these firms really considers local caching. In today's world, if you buy a book or a record, you can use it as many times as you like, for the one fixed fee, but the conglomerate vision of the future will have you downloading the material every time, and paying a fee every time.

On the other hand, the Internet has been run on a fixed-price basis for so long that an entire culture has grown up around the mythical notion of free access. In truth, no access is free – even if you don't pay for it directly (most companies do pay for their connections directly), you just pay for it indirectly. Even though there is no such thing as a truly free net connection, it is mostly true that one can obtain a certain amount of Internet connectivity at a flat fee for a given level of service. This is an important concept, because usage sensitivity is not a part of the traditional net access pricing formula.

A question comes to my mind: what information is going to be available? And even more importantly: what is it going to cost? I'm no more into paying per access or per byte than anyone else is. On the other hand, I certainly understand that capitalism is what we've got for an economic system, and it more or less works. So, given that, what sorts of things are reasonable?

I think that one of the most important factors is going to be the question of comparable value. For example, right now I can spend $69 on a cheap TV, and get all of the over-the-air broadcast stations I can stand for a nominal direct cost in electricity. For a little over two hundred dollars more, I can buy a better set and a rooftop antenna, and again for practically no direct recurring cost I can bring in even more stations, even from distant cities. I don't remember the figures for penetration of cable into US households, but there are plenty of holdouts for "free" TV, including my own household. The point is that there are lots of folks who don't want to pay for something they see out there free for the taking.

Take another example. In most of the US, you can rent a residential phone line with flat-rate local service. You can call any local number you want, as many times as you want, for as long as you want, and you don't pay anything but a flat fee. Now, there are telco folks who will argue that the fixed-rate lines lose money (and there are places in the US, like Chicago, where you can't get flat-rate service), but that doesn't cut any mustard when individual consumers make cost comparisons. The average consumer, when pricing information services, isn't going to be as interested in those services if they come with per-minute charges. Sure, there will always be a place for a CompuServe, with its large value-add, but look at the growth rate of America OnLine, with a semi-flat-rate service. You could argue that flat-rate services have response-time problems, but I believe that is more a function of the monolithic and centralized computing structure.

One final example. Each week, I receive a news magazine in the mail. This news magazine is chock-full of high-resolution color pictures and a few tens of articles. The direct cost to me for this service is on the order of a dollar a week.

The point of all of this? The current cost of information in various formats delivered to my home is actually incredibly low. When all of this information is available online, the cost is going to have to be comparable to the current physical world cost, or the service is not going to be viable economically. I, and many other people, do not wish to pay a penny a byte, or $5 an hour, or other high rates, for the same information that I can get off-line for a fixed, and much lower, fee. Some other added value would be required to make the online service worthwhile. For that matter, an Educom item in the December issue of *;login:* cited a study of Mac users, in which three-quarters of them said that they would not pay $10 per month for an online service.

Frankly, I rather agree with them, and I ran a tiny service that charged $10 per month.

For most people, the value just isn't going to be there. Those large corporations I mentioned are betting billions of dollars that my point of view is wrong. However, they're going to have a tough row to hoe, as they are going to be competing with "free" broadcast TV, as well as the existing cable systems, libraries, and magazines.

They're also going to have to compete with video stores. How many $2 movies are they going to have to rent over their high-speed networks in order to pay for them? Don't forget, they won't achieve a high penetration unless their line rentals are low and relatively fixed. They may be able to make some headway by providing things that aren't currently broadcast, like software, but that actually gets them out of the content production business, and back into a more purely transport-oriented role. Few people will pay a premium for software over the network, so the transport costs would have to be roughly comparable to media costs, which are cheap and getting cheaper.

There is also the small matter of equipment costs. On the one hand, the Internet model puts much of the costs on the consumer. This would be good for the information providers, because it helps to reduce their capital investment. On the downside, it will also limit penetration, by making it difficult and costly for customers to equip themselves to reach the entry point for the services.

On the other hand, the large corporation model uses the television as the point of presence of their service, and equips it with a so-called "set-top box," which is essentially a special-purpose computer. The set-top box is intended to be cheap to produce in volume, so that the cost to the customer, if the boxes are sold, or the provider, if they are rented, is low. There will be an ongoing tension between network providers, such as cable and phone companies, and computer hardware and software vendors, as the two factions slug it out. Perhaps the deciding factor in which model comes to dominate the scene will be the continuing penetration of personal computers into households. If the computer firms can leverage this access, they can make the computer the access method of choice. If not, the set-top boxes will come to proliferate, and the overall picture will not look like an Internet.

The dominating factor for the growth of online information is going to be cost. Unless the costs and benefits are comparable to existing media, people just won't pay the freight. However, there is one tremendous potential mitigating factor in this cost picture. Right now, many hours of mass-media programming are paid for by advertisers. I'm sure that nobody on the Internet wants a net feed that they pay for "polluted" with advertising sent by companies who view it as a free medium. But what if the cost picture were turned around? What if advertisers paid or subsidized you to read their ads? I would absolutely review a small amount of advertising daily if some advertiser would pick up the cost of my net feed, and I'm sure millions of consumers would do likewise, since they already watch (or ignore) commercials anyway. The advertisers might want to guarantee you actually view the ad, of course, but at least the choice would be up to you.

The one factor that could potentially fuel the explosion is one that the big firms have so far ignored. They would like to sew up the provision of information and the transport in one neat package. However, (and I could be wrong) I think that the opportunity to provide information is part of what is fueling the growth of the World Wide Web, and I think this could translate into the future network. In order for this to work successfully, the network will have to support a system for exchanging money. It will not be pay-per-byte, or per-picture, but there will be payments nonetheless. Someone who pays a dollar a week for a news magazine is not going to pay a nickel a page and ten cents a picture to see the same information over the network, but would probably pay the same dollar a week to access a comparable source of information.

Furthermore, even the average consumer, who claimed that he/she wouldn't pay any extra to watch a movie over the network that they can rent at the corner video store, would most likely take advantage of grass-roots expertise. The big companies aren't going to be able to, or necessarily want to, provide impartial analyses of products, but you can already find some on the net. For example, I'm sure *Consumer Reports*' auto price information would do splendidly online, as would their other reviews. I would gladly forgo the paper copy, but still pay my annual membership fee, to have online access to their information, even if I only used it rarely.

The big firms aren't going to provide specialists in a million niche fields, but they'll reside on the net, and with the net you could find them and hire them. The net is going to supply many opportunities for skilled individual or corporate entrepreneurs. Opportunities will continue to evolve for folks who act as gatekeepers, moderators, and editors; they will be compensated for their time and effort.

Perhaps the one place the large firms really have it right it is home shopping. Certainly, the enormous success of the Home Shopping Network and QVC indicate that. Even now, Internet-based shopping is springing up. I think even there, though, the model is not going to be the Mall of America, but rather the specialty store and the neighborhood shops. On the Web, there is no virtue to size; what is important is accessibility, and that will come with organization by type. I would rather shop for computer equipment from a page listing all of the sellers on the net, so I could compare prices.

I expect someone may be able to make a nice living indexing and tracking prices and product information. For other items, I might want to shop in my physical locale, so that the delivery charges are reasonable (you probably haven't ordered a pizza over the Web unless you live in Santa Cruz).

The difference between the corporate model and this one is that the information providers can be anyone, not just the network conglomerate. I personally could periodically publish a "magazine" of my thoughts, and charge a small amount to each subscriber, not per access or per byte, but just for the right to make and view a personal-use copy, just as a magazine subscriber can do now. Of course, there's no reason to believe anyone would buy it.

Whatever way it goes, things are going to be very different, and we all are going to see some great changes. There are a lot of very large bets being put on the table, and the odds are not at all as clear as some of the players think.

# TCP/IP Addressing Strategies for Switched Networks

*by Frank Crone*
*<Frank.Crone@amd.com>*
*and John Jarocki*
*<John.Jarocki@amd.com>*

## Abstract

Switched local area networks are beginning to replace router-based subnetted LANs. To date, products are available that are capable of switching (or bridging) multiple ethernet, FDDI, and Token Ring network segments. As the transition to switched networks is being made, many network administrators are faced with the challenge of combining hosts that were once distributed across many subnets into a single switched network. This paper presents several different methods for addressing network hosts using multiple IP subnetwork addresses on a single MAC-layer switched network.

This report focuses primarily on the networking environment within AMD, but each of the options presented here have relevance in other environments as well. As a result of this internal focus, the scope of this paper has been limited to TCP/IP addressing issues only, even though some areas of AMD use other network protocols such as DECnet and NetBeui. The closing section of this report presents a few recommendations for addressing TCP/IP hosts in switched networks. These recommendations should be followed where possible to maximize the performance TCP/IP hosts on switched LANs.

## I. Introduction

Routers have evolved as the dominant building block for designing large internetworks. However, the use of segmentation and subnetting via router interfaces is no longer scalable and manageable enough to support many of today's networking requirements. As system performance has grown, router-based ethernet segments have been forced to grow smaller, meaning that a larger percentage of network traffic on each network segment is actually destined for hosts on other subnets.

A relatively new type of networking device, the switching bridge (Ethernet/FDDI/Token Ring Switch), has evolved to meet the demand for small network segments with transparent intra-segment communications. The appearance of these switches has allowed groups that have been divided between several different TCP/IP subnetworks (subnets) and router interfaces to be combined into a single, homogeneous switched network. It has also provided a mechanism for groups using single subnets to scale their network size and performance.

Several factors must be considered when combining more than one existing subnet into a single switched network. These include segment size, host/segment grouping, and host addressing. This paper will present several different methods for addressing hosts when using more than one IP subnetwork address in a single switched network. The pros and cons of each method will be discussed, and the performance differences between each method will be illustrated.

## II. Fundamentals of Switched Ethernet/ FDDI Networks

Ethernet/FDDI switches forward packets between ports based on the hardware, or MAC (Media Access Control) address of the hosts connected to it (just like ethernet bridges in the past). The switch keeps a table of all MAC addresses it receives and the corresponding port for each address. When the switch receives an ethernet or FDDI frame, it reads the destination MAC address, checks the switch's bridging table, and forwards the packet to the appropriate port. Since a packet switch operates at the MAC layer as opposed to the network layer (i.e., ethernet/FDDI protocols rather than TCP/IP or NetBIOS protocols), broadcast packets for an IP subnet will be seen on all ethernet and FDDI segments that have hosts in that IP subnetwork.

Packets directed to a particular IP address, however, will only be seen on the sending and receiving segments. The sending host uses the ARP protocol to look up the MAC address associated with a particular IP address and sends the packet to that MAC address. Since switches operate independent of Network Layer protocols, hosts are not required to rely on a network layer router to communicate across segments. Because routers add significant processing latency to every packet they forward, the overall network response time can be improved by using a MAC layer switching device.

A typical switched Ethernet/FDDI network is shown in Figure 1. Two or more switches are interconnected via an FDDI ring that also provides connectivity for high-powered servers and in some cases to an external gateway or default router. Each switch provides several switched ethernet ports. These ports are used to create shared segments and to provide dedicated 10 Mbps Ethernet connections to servers. Each port on a switch provides an individual collision domain; traffic destined for one switch segment is not seen by hosts on another segment. Only broadcast packets are seen by hosts on all segments. (Network Layer broadcast packets are translated into MAC-layer broadcasts. Switches will forward a MAC-layer broadcast to all ports by default.)

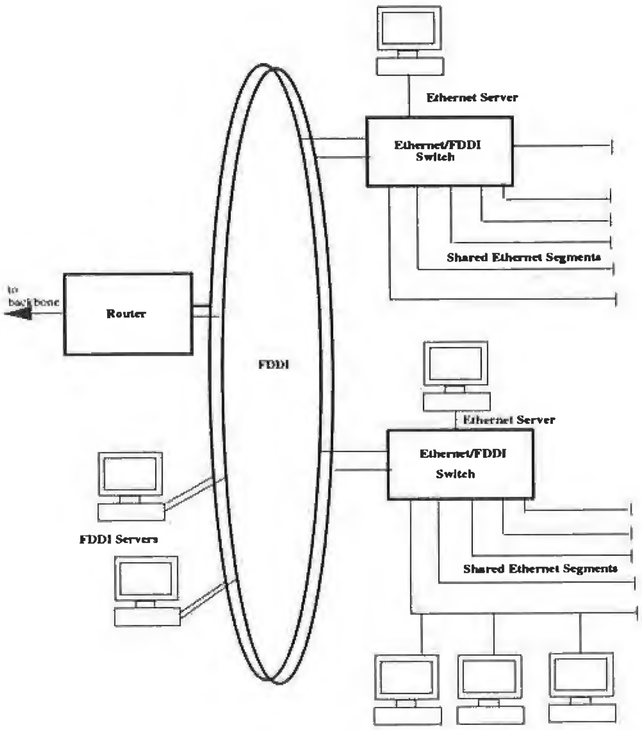


**Figure 1. Typical Switched Ethernet/FDDI Network**

Ethernet/FDDI frame switches must be able to translate packets between the different network media they support. Packets that are forwarded onto the FDDI ring from an ethernet segment are translated into FDDI packets (called "translational bridging"), and visa versa. This allows hosts on the FDDI ring to take full advantage of the large MTU of FDDI, while allowing hosts on the ethernet segments to communicate directly to the router and to servers on the ring. Switches that are capable of fragmenting large FDDI frames into smaller ethernet frames achieve what is called "IP Fragmentation". Translational Bridging and IP Fragmentation enable seamless integration of the client/server architecture in a switched ethernet/FDDI environment.

## III. Addressing Options

Switched Ethernet/FDDI networks are easy to implement when using a single IP network or subnetwork address. Each host is connected to one of the switch segments, and the default router is connected to either a dedicated ethernet port or to the FDDI ring. However, there are many reasons for a group to wish to use more than one IP subnetwork (subnet) address when implementing a switched network. First, any network of more than 254 hosts will require more than one class C subnet address. Also, many groups are already divided across more than one class C IP subnet and they may prefer not to change existing addresses.

Another alternative for adding more than 254 hosts to a single subnet is to use variable length subnet masking. Using this approach, the subnet mask can be changed to 255.255.254.0, which will allow over 500 hosts within a single subnet. However, not all routers or hosts can support variable length subnet masks, so the remainder of this paper will focus on traditional netmasks that fall on octet boundaries.

Since ethernet/FDDI switches forward packets based on the MAC address of the host, combining more than one class C subnet in a switched network requires some forethought. The reason is if a host in a switched network has an IP address of 163.181.x.1 and a subnet mask of 255.255.255.0, it will not communicate at the MAC layer with another host on the same network that has an address of 163.181.y.1 and a subnet mask of 255.255.255.0. In this case, each host would think that the other was on a different network segment, and all packets exchanged between them would be sent to the default router to be forwarded at the network (IP) layer. While this addressing strategy may be used in some scenarios, it does not exploit low latency forwarding capabilities of the switch. This strategy, along with two alternatives, are discussed fully in the sections below.

### Subnetted Class B Address with Class C Netmasks

This addressing option was discussed briefly in the section above. A typical implementation of this addressing scheme is shown in Figure 2. Since each host has a subnet mask of 255.255.255.0, communications between hosts with subnet x and y addresses will be forwarded through the default router. In this case, the router interface has a primary address of 163.181.x.99 and a secondary address of 163.181.y.99.

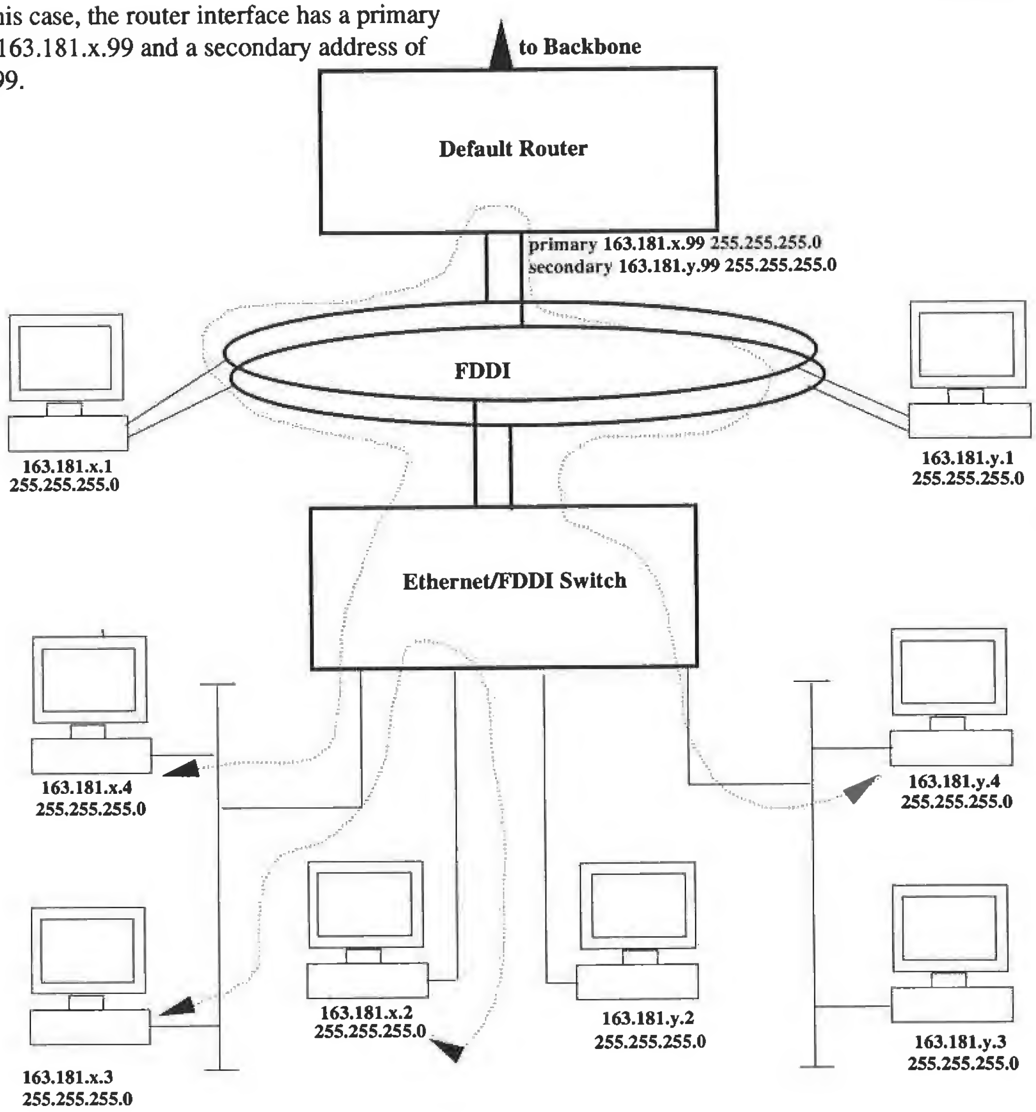


**Figure 2. Using Subnetted Class B Addresses with Class C Netmasks**

Only traffic between hosts using the same subnet addresses will benefit from the low latency of the switch in this scenario. As shown in this diagram, this traffic is forwarded at the MAC layer through the switch.

The benefits of using this addressing strategy are:

- More than one group can use the switch. By separating the groups into multiple class C subnets, a single switch can support a shared switched network. One drawback to doing this, however, is that each group will be subjected to the MAC layer broadcast traffic of the other groups sharing the switch.
- For workgroups that are already split across more than one IP subnet, this option will create the least amount of maintenance overhead when migrating to a switched network. All of the hosts involved can retain their current IP address and subnet mask.

The detriments of this addressing strategy are:

- Communications between hosts with different subnet addresses will be forwarded through their default gateway. This does not take advantage of the low latency of the switch, and places undue traffic on the router interface.
- The link between the switch network and the router will become unnecessarily "hot" as packets are U-turned at the router interface to cross switch segments. Traffic destined to other networks will have to contend with traffic that should have remained local to begin with.

- Packets that are exchanged by hosts with different subnet addresses will have to be processed by the switch twice: once as the transmitting host forwards the packet to the default router, and a second time as the router forwards the packet to it's destination.

A complete analysis of the performance of this addressing strategy is presented later in this paper.

## Class B Address and Class B Netmasks

Ordinarily in a class C subnet like 163.181.3.0, only addresses that begin with 163.181.3 will be considered "local" and hence ARP'd for. If you use a class B netmask where class C subnets are in effect, any address beginning with 163.181 will generate an ARP request, but only those hosts not separated by a router "hop" will respond directly. If the router is using the Proxy ARP protocol, however, it can respond in the name of any remote IP addresses by responding to the ARP with its own MAC address. Then it will receive packets destined for that IP address and forward them to the correct remote segment.

This addressing scheme eliminates most of the problems discussed above. As shown in Figure 3, network hosts can use various ranges within a class B network address. Unlike

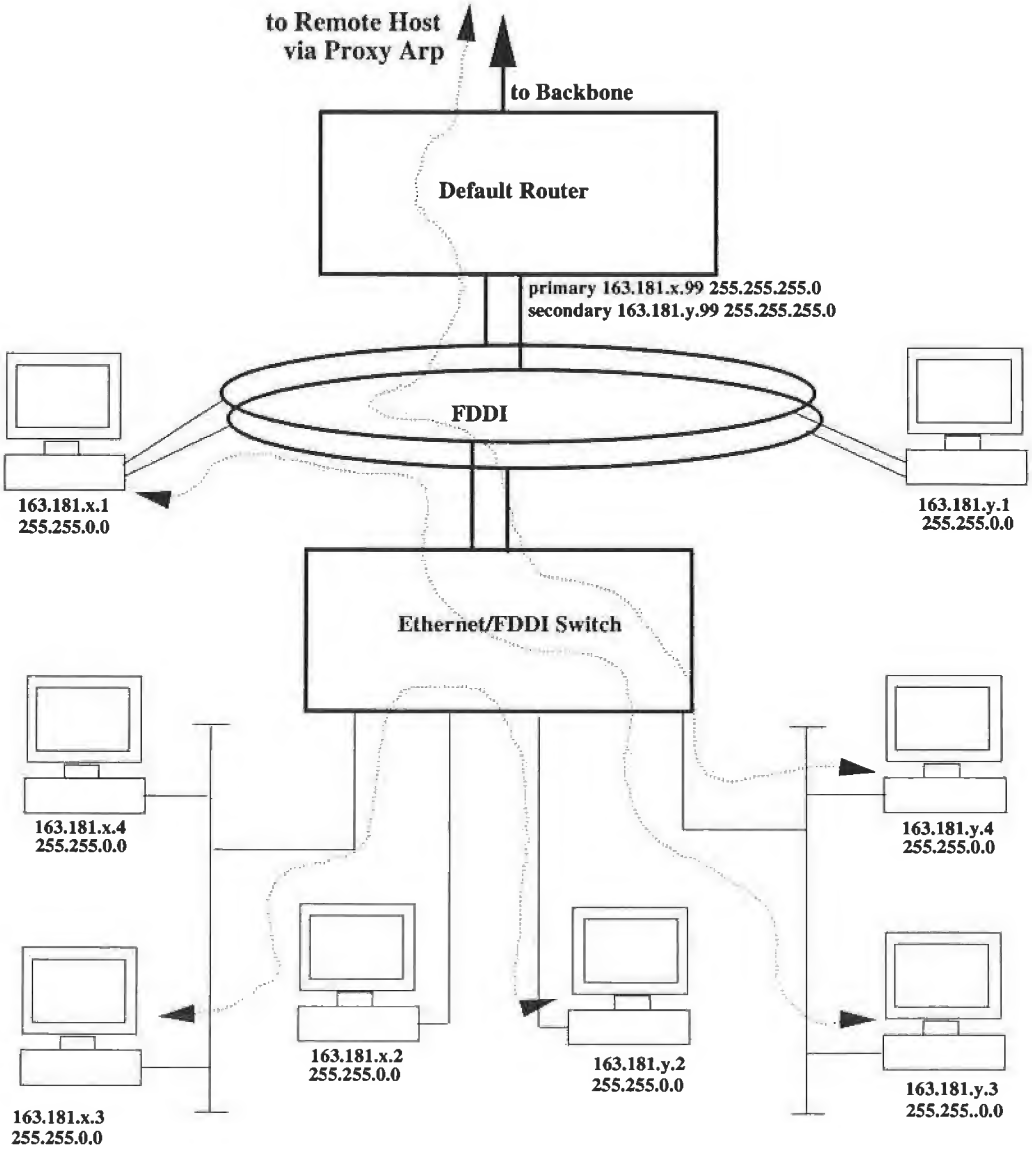


Figure 3. Using Multiple Class C Addresses with Class B Netmasks

the previous example, the subnet mask for each host is changed to a class B netmask (255.255.0.0). The router is assigned a class C subnetted primary address and as many secondary addresses as needed. In this case (figure 3), each host will see all 163.181.0.0 addresses as local, and will ARP to obtain the MAC address of any destination host in that domain. When the destination host is on the local switched network, it will respond to the ARP and the two hosts can communicate directly through the switch using hardware addresses. When the destination host is on a remote network, the router will respond to the ARP using the Proxy ARP protocol. The local host will then forward all packets to the router using the router's MAC address just as if the remote host had responded directly.

The benefits of using this addressing strategy are:

- Communications between hosts with different subnet addresses will be accomplished via the switch using MAC addresses.
- Hosts can keep their existing IP addresses.
- As a host moves to other segments within the switched network, its address will not have to change. There will be no distinction between the different class C addresses while the subnet mask is set to a class B.

The detriments of using this addressing strategy are:

- The subnet mask on each host will have to be changed to 255.255.0.0 if it was previously configured with a subnet mask of 255.255.255.0.
- All gateways on the switched network must be capable of supporting the Proxy ARP protocol. This protocol is not supported on some gateways, such as Apollo gateways. This means that hosts in a switched network using this addressing strategy will not be able to communicate with Apollo Domain hosts through an Apollo gateway if the Apollos also have a 163.181.0.0 network address.

A complete analysis of the performance of this addressing strategy is presented later in this paper.

## Class C Subnets and Class C Netmasks with Static Routes

The previous two sections of this paper have discussed methods for addressing hosts using more than one IP subnet within a switched network. Each of these methods has been shown to have several advantages and disadvantages. The option presented in this section alleviates most of the major disadvantages discussed with the previous methods.

So far, the best alternative presented for combining hosts on multiple subnets was to change their subnet mask to 255.255.0.0. However, this option was shown not to be viable in the case where certain gateways on the network do not support the Proxy Arp protocol. What is needed in this case is an addressing strategy that allows hosts to communicate across IP subnets at the MAC layer without loosing visibility to other networks that are connected via a gateway that does not support proxy arp.

The solution to this problem is to assign a secondary, static route on each host within the switched network. Using this scheme, each host is assigned a class C address and netmask. Instead of using only one default gateway, however, a special entry is made in the host's routing table to make all subnets within the switched network appear to be locally connected networks. This is accomplished by editing the `/etc/rc.local` file on Sun workstations or the `/etc/netlinkrc` file on Hewlett-Packard workstations. The updated entries in these files have the form:

```
/etc/route add net 163.181.y.0
      'hostname' 0

/etc/route add net 163.181.z.0
      'apollo-gateway-name' 1

/etc/route add default 163.181.x.99 1
```

In first line of this example, 'hostname' refers to the name of the local host, while subnet 163.181.y.0 is a second subnet used within the same switched network. This line effectively tells the local host that it is the router to that network so all packets destined to that network can be forwarded directly by the local host, and not sent to the default router. The second line defines a route to an Apollo network supported by an Apollo gateway named by 'apollo-gateway-name'. The third line assigns the default router address for a host on subnet 163.181.x.0. This line should already be present on all hosts. This addressing strategy is illustrated in Figure 4 on the following page.

The advantages in using this addressing scheme are:

- Hosts within a switched network supporting more that one IP subnet can communicate through the switch at the MAC layer without losing visibility to other networks (such as Apollo token-ring networks).
- Hosts within the switched network do not have to change their addresses or netmasks if they are already configured with a class C netmask.
- Hosts within the switched network can still communicate (via the default router) while their routing tables are being updated, eliminating unnecessary downtime.

The drawbacks to using this addressing scheme are:

- Personal Computer networking applications typically do not support multiple route entries per interface. This means that inter-subnet PC traffic within a switched network will still be forwarded to the default router.
- Each workstation within the switched network will need to have its routing tables updated manually by the system administrators.
- It may be hard to determine if a host is configured incorrectly (without secondary route entries) since an incorrectly config-

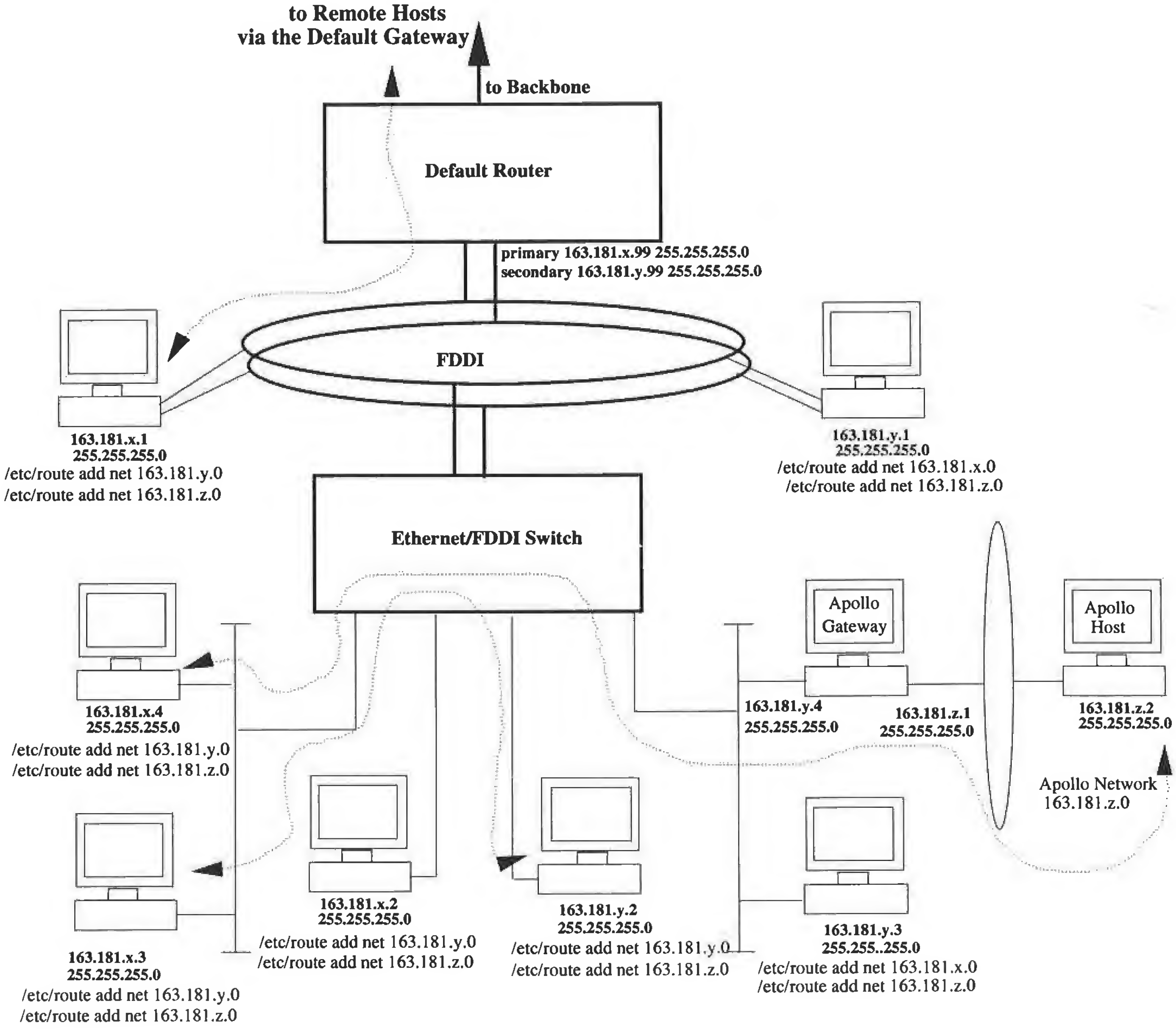

to Remote Hosts
via the Default Gateway
to Backbone
Default Router
primary 163.181.x.99 255.255.255.0
secondary 163.181.y.99 255.255.255.0
FDDI
163.181.x.1
255.255.255.0
/etc/route add net 163.181.y.0
/etc/route add net 163.181.z.0
163.181.y.1
255.255.255.0
/etc/route add net 163.181.x.0
/etc/route add net 163.181.z.0
Ethernet/FDDI Switch
Apollo
Gateway
Apollo
Host
163.181.x.4
255.255.255.0
/etc/route add net 163.181.y.0
/etc/route add net 163.181.z.0
163.181.y.4
255.255.255.0
163.181.z.1
255.255.255.0
163.181.z.2
255.255.255.0
Apollo Network
163.181.z.0
163.181.x.2
255.255.255.0
/etc/route add net 163.181.y.0
/etc/route add net 163.181.z.0
163.181.y.2
255.255.255.0
/etc/route add net 163.181.y.0
/etc/route add net 163.181.z.0
163.181.y.3
255.255..255.0
/etc/route add net 163.181.x.0
/etc/route add net 163.181.z.0
163.181.x.3
255.255.255.0
/etc/route add net 163.181.y.0
/etc/route add net 163.181.z.0

**Figure 4. Using Secondary Network Routes on Host Workstations**

ured host can still communicate across subnets through the router. The only obvious indicator of a misconfigured host would be poor network performance as perceived by the user of that host.

## IV. Performance of Addressing Options

The previous sections of this paper have described several options for addressing switched ethernet/FDDI networks that combine more that one class C IP subnet. This section will present performance data for each option as determined through laboratory testing.

### Testing Method

*Measurement Tool*

The tool used for measuring the network performance of each addressing strategy was the network benchmark application "Netperf". This UNIX-based application is capable of measuring both TCP and UDP stream throughput as well as TCP and UDP request/response rates between two UNIX workstations. For each addressing scheme, a series of stream and request/response tests were run to measure the effectiveness of each configuration.

*Hardware Configuration*

Each test was performed using the hardware configuration shown in Figure 5. The netperf application tests were run using a Sun Sparc ELC and a Hewlett-Packard 715. As shown in this figure, the two workstations were connected to switched ethernet ports on an SMC ES/1 ethernet/FDDI switch. The switch was then connected to a Cisco AGS+ router using a Synoptics 3030 concentrator. This configuration was changed as shown in Figure 6 to re-test the addressing strategy with the worst performance. As illustrated in this diagram, the connection between the switch and the router was changed to FDDI.

*Host Configuration*

The host configurations for each series of tests are listed below.

- Baseline – each host was addressed for the same class C subnetwork with a class C subnet mask. This configuration theoretically represents the ideal addressing scheme. All other test results are compared to this configuration

- add_c_mask_c – the two hosts were addressed using different class C subnetworks with class C subnet masks. This configuration should theoretically have the worst performance characteristics.

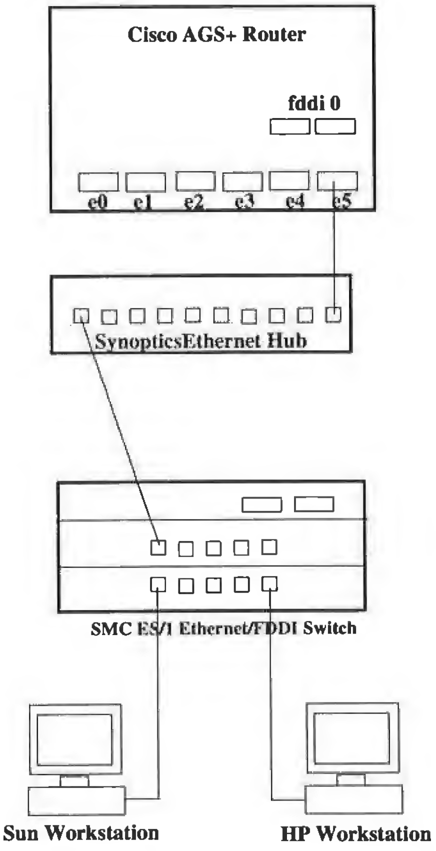


**Figure 5. Laboratory Hardware Configuration 1**

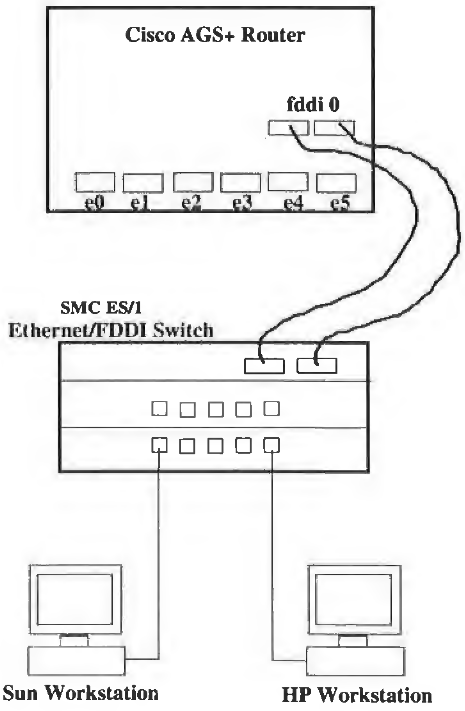


**Figure 6. Laboratory Hardware Configuration 2**

- add_c_mask_b – the two hosts were addressed using different ranges of the same class B address with class B subnet masks.

- add_c_static – each host was configured using different class C subnetworks with class C subnet masks. A secondary entry was then made on the host's routing tables to localize the different subnets.

- add_c_mask_c_fddi – same as number 2 using the hardware configuration shown in Figure 6.

*Test Suite*

The various netperf tests used for each host configuration are listed below.

- UDP Stream – a steady stream of UDP packets are sent from one workstation to the other for 20 seconds. The test is repeated for packet sizes of 64, 128 and 1024 bytes.

- TCP Stream – a steady stream of TCP packets are sent from one workstation to the other for 20 seconds. The test is repeated for packet sizes of 64, 128 and 1024 bytes.

- UDP Request/Response – A series of UDP requests are generated by a workstation for 20 seconds. The receiving workstation then replies to each received request with the appropriately sized response packet. The test is repeated for request/response packet sizes of 1/1, 64/64, 128/128 and 128/1024 bytes.

- TCP Request/Response – Same as above except using TCP.

A summary of each configuration and test suite is shown in Table 1.

Table 1: Test Configurations

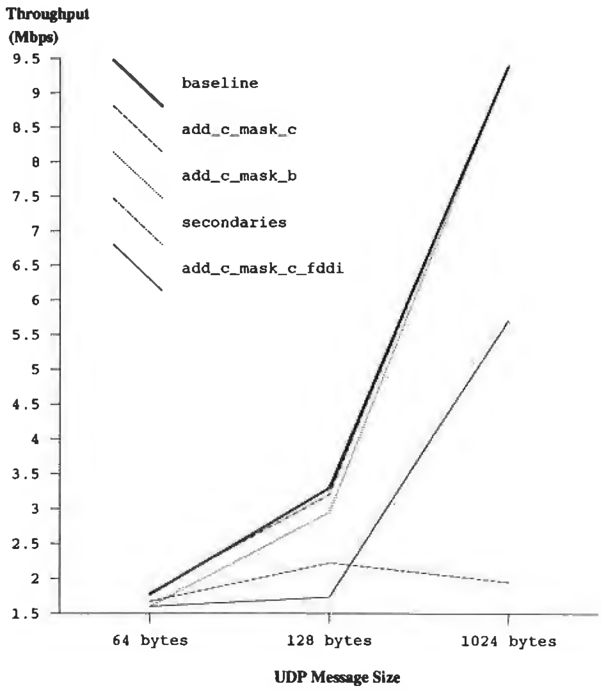
| | UDP_Stream | TCP_Stream | UDP_Req/ Resp | TCP_Req/ Resp |
|---|---|---|---|---|
| Hardware Cfg 1 | baseline | baseline | baseline | baseline |
| Hardware Cfg 1 | add_c_mask_c | add_c_mask_c | add_c_mask_c | add_c_mask_c |
| Hardware Cfg 1 | add_c_mask_b | add_c_mask_b | add_c_mask_b | add_c_mask_b |
| Hardware Cfg 1 | add_c_static | add_c_static | add_c_static | add_c_static |
| Hardware Cfg 2 | add_c_mask_c | add_c_mask_c | add_c_mask_c | add_c_mask_c |

## Results

The results from the laboratory testing have been broken into four groups based on the type of test that was performed. The four groups include: UDP stream, TCP stream, UDP request/response, and TCP request/response. For each test group, the performance of the various addressing strategies are compared against the baseline results. An analysis of these results is presented in Section V of this paper.

*UDP Stream Test Results*

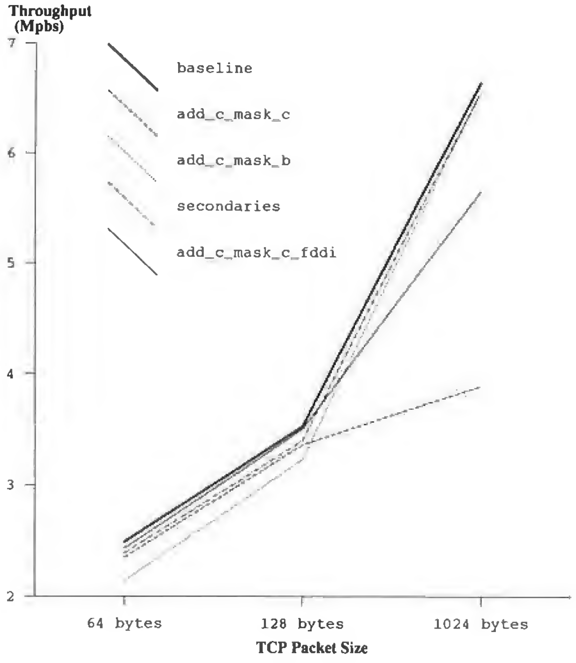
A summary of the UDP stream test results is shown in Graph 1. The horizontal axis of this graph represents the packet size generated by the transmitting workstation, while the vertical axis represents the amount of data received by the receiving workstation.



**Graph 1. UDP Stream Test Results**

*TCP Stream Test Results*

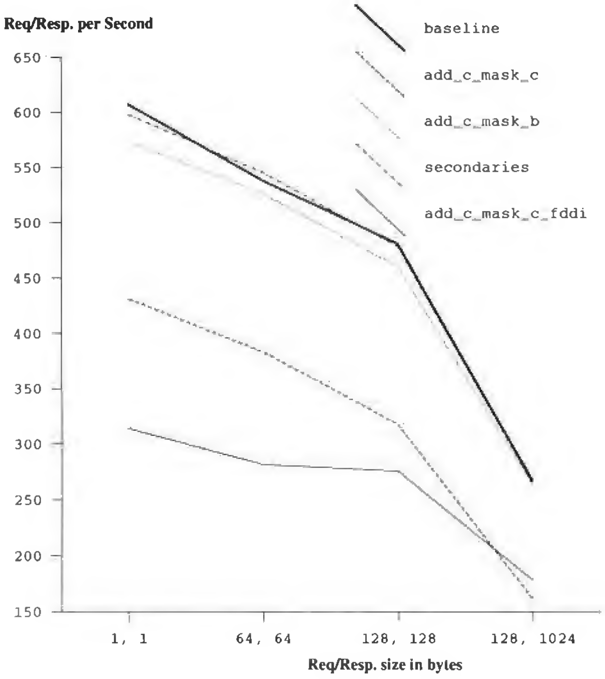
A summary of the TCP stream test results is shown in Graph 2 on the next page. The layout of this graph is the same as Graph 1.



Graph 2. TCP Stream Test Results

*UDP Request/Response Test Results*

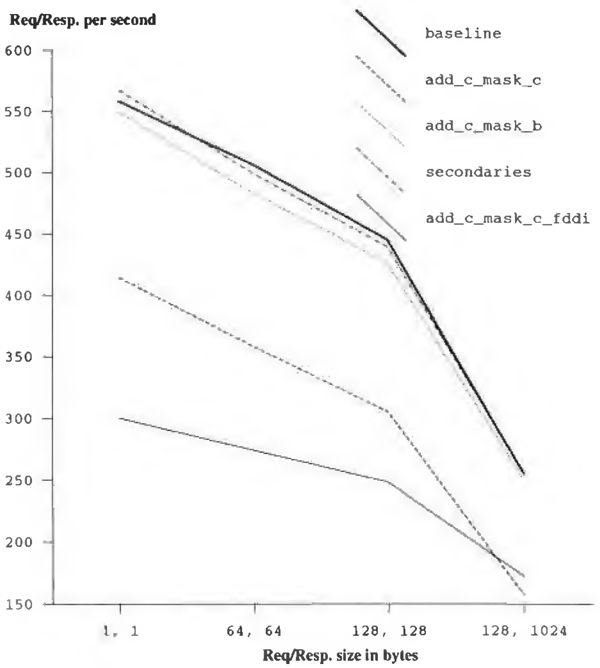
The results from the UDP request/response tests are illustrated in Graph 3. The horizontal axis of this graph represents the request/response message size. The vertical axis represents the number of transactions per second for a particular message size.



Graph 3. UDP Request/Response Test Results

*TCP Request/Response Test Results*

The results from the TCP request/response tests are shown in Graph 4. This layout of this graph is the same as the udp request/response graph (Graph 3).



Graph 4. TCP Request/Response Test Results

## V. Analysis

The previous section of this report, along with the associated graphs, have given the results of several benchmark tests using each addressing strategy. This section will analyze the results for each suite of tests.

### Baseline Configuration

As shown in Graphs 1-4, the baseline configuration had the best overall performance characteristics of all addressing options. In each graph, the baseline result shows the ideal performance characteristics for a single subnet switched ethernet/FDDI network.

### Class C Addresses and Class C Netmasks with Static Routes

The test results shown in Graphs 1-4 show that this addressing strategy performed nearly as well as the baseline configuration. Even though each workstation was addressed for a separate class C IP subnet, the additional static route entries achieved the desired performance levels by eliminating packet processing by the router.

### Class C Addresses and Class B Netmasks

This addressing strategy was shown to perform slightly below the baseline and static route addressing strategies in Graphs 1-4. However, this strategy should theoretically have the same performance as the static route addressing strategy. Further analysis on this result has shown that a combination of Proxy Arp traffic and a remotely mounted file system (on the Sun ELC) caused enough additional network traffic to slightly shift the test results. Subsequent testing performed with no remotely mounted filesystems on the Sun ELC have shown that the results from this configuration will be the same as the static route configuration as long as there is no off-net traffic. However, real world networks will never be clean of all off-net traffic due to applications such as e-mail, news services, etc. While this type of traffic cannot be avoided, the small amount of Proxy Arp traffic generated by the router can be avoided by using the static route addressing strategy.

### Class C Addresses and Class C Netmasks

As expected, Graphs 1-4 show that this configuration does not perform nearly as well as the preceding configurations. In fact, Graph 1 shows that the performance of this option actually degrades as the amount of UDP stream traffic is increased. There are several reasons for the exceptionally poor performance of this addressing strategy, including:

- Additional processing by the switch as each packet is processed twice: once from the transmitting host to the router and once from the router to the destination host.
- Collisions on the router segment caused additional retransmits and additional packet processing by both the switch and router.
- Dropped packets by the router. Since the router was simultaneously receiving and transmitting packets on the same interface, many packets were dropped from the input and output queues on the router.

### Class C Addresses and Class C Netmasks with FDDI

Graphs 1-4 show that this addressing strategy and hardware configuration actually had the worst overall performance of all tests. Even though the link from the switch to the router was changed from 10Mbps ethernet to a collisionless 100 Mbps FDDI ring, the performance of this solution was still worse than any other configuration. This poor performance was caused by the processing overhead associated with translating ethernet packets to FDDI and visa versa. While the effects of this processing are minimized when transmitting large packets, they are very prevalent when transmitting small packets. When added to the additional processing that must be performed by the router using this addressing scheme, the total effect will minimize the total end to end throughput of any switched network. This is a good example of how an unnecessary, unplanned implementation of additional bandwidth can actually degrade the performance of a switched ethernet/FDDI network.

## VI. Recommendations

The previous sections of this paper have discussed the usage and performance of various TCP/IP addressing strategies in switched ethernet/FDDI networks. This section will attempt to summarize the results in the form of recommendations for addressing hosts on switched networks. Unfortunately, the best method for addressing a particular network depends on too many variables to make a single statement as to which type of addressing is the best. For that reason, the recommendations in this section should be taken as a loose guide for addressing switched networks based on several contingencies, and not as an absolute reference as to which method should be used in all switched networks.

It is worthwhile to note that more than one type of addressing scheme can be used on host machines in a switched ethernet/FDDI environment. In most real world networks, this will probably be the rule, not the exception. In most cases, the type of addressing configuration used will depend on the platform it is being used on, such as a PC, Sun workstation, HP workstation, or any other type of network node such as an ethernet-based Apollo workstation. For this reason, the following recommendations are being divided according to the hardware platform that will be using them.

### UNIX Workstations

UNIX workstations, such as Sun SPARCstations and Hewlett-Packard 700/800 series, offer the greatest flexibility in selecting an addressing scheme. Based on the test results presented in Graphs 1-4, and on the ease of implementation, it is recommended that UNIX workstations be addressed using additional "route add"commands in the appropriate configuration file. This option appears to be the best method for implementing multiple IP subnets in a switched environment for the following reasons:

- Implementation is easy on existing workstations. "Route add" commands can be done on a UNIX workstation without the need for a reboot, and the permanent configuration change can be made by updating only one configuration file.

- Visibility of all networks is maintained. Since this configuration does not rely on proxy arp, routes through gateways that do not support this protocol can be maintained easily.
- The additional network traffic generated by the proxy arp protocol is avoided.

### Personal Computers

While UNIX-based workstations offer the greatest addressing flexibility in switched networks, personal computers are perhaps the most difficult. TCP/IP implementations for PCs do not typically offer advanced features such as adding secondary static routes or using multiple gateways. This leaves two options: address them for different subnets using a class C subnet mask, or address them for different subnets using a class B subnet mask. If possible, it is highly recommended that the PCs be addressed with a class B netmask using proxy arp for off-net communications. As discussed in Section III.B, this option will not work for networks using gateways such as Apollo Domain gateways that do not support proxy arp. In this case, the PCs will have to be masked at 255.255.255.0, and they will have to rely on the default router for cross subnet traffic within the switched network.

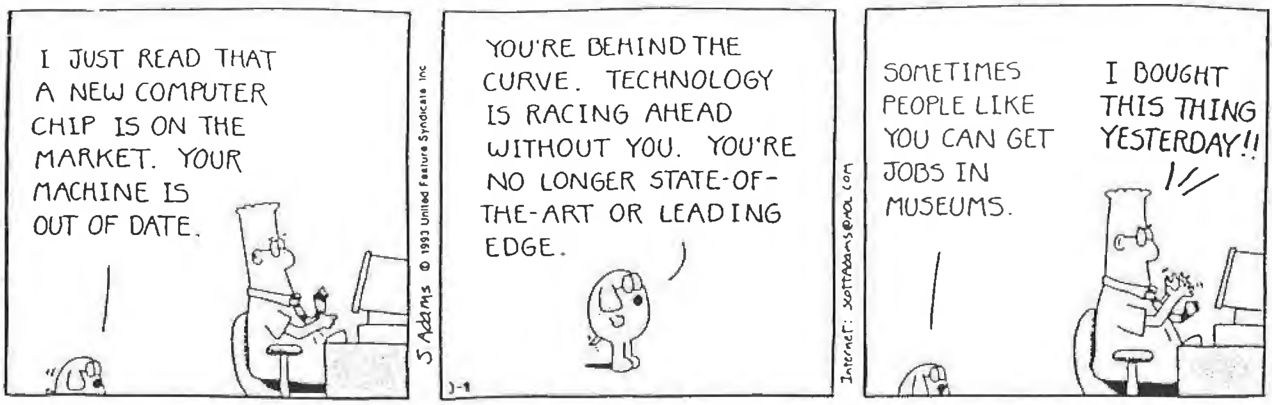
### Workstations Without Secondary Routes

Some older workstations do not allow "multiple logical subnets" to be assigned to a network interface, meaning that additional routes can not be added for additional class C subnets. An example of this is an Apollo workstation (running Domain/OS). Ethernet based Apollo workstations in a switched environment are much like PCs in this case. If there are no other gateways on the switched network except for a default router, then they should be configured with a class B netmask. However, if they must communicate to hosts via some other gateway, they will be required to use a class C netmask.

### VII. Conclusion

This report has presented and analyzed several different methods for using more than one class C subnet address in a switched ethernet/FDDI environment. The goal was to determine which addressing scheme would perform as close as possible to a configuration using only one class C subnet. Two methods were found that offer comparable performance. However, one option--adding static routes to each host--was shown to be a slightly cleaner implementation that using a class B (255.255.0.0) netmask on each host. Regardless, the final decision on which approach to take still depends on the environment of the switched network and on the host platform being used.

**DILBERT® by Scott Adams**


I JUST READ THAT A NEW COMPUTER CHIP IS ON THE MARKET. YOUR MACHINE IS OUT OF DATE.
S Adams © 1993 United Feature Syndicate Inc
YOU'RE BEHIND THE CURVE. TECHNOLOGY IS RACING AHEAD WITHOUT YOU. YOU'RE NO LONGER STATE-OF-THE-ART OR LEADING EDGE.
3-1
Internet: scottadams@aol.com
SOMETIMES PEOPLE LIKE YOU CAN GET JOBS IN MUSEUMS.
I BOUGHT THIS THING YESTERDAY!!

DILBERT reprinted by permission of UFS, Inc.

# An Update on Standards Relevant to USENIX Members

*by Nick Stoughton*
USENIX *Standards Report Editor*
*<nick@usenix.org>*

The following reports are published in this column:

- •POSIX.1:System Interface
- •POSIX.4: Realtime Interfaces
- •A View from the Chair

Our Standards Report Editor, **Nick Stoughton**, welcomes dialogue between this column and you, the readers. Please send your comments to *<nick@usenix.org>*.

## Report on POSIX.1: System Interface

*Shravan Pargal <pargal@sdiv.cray.com> reports on the October 17-21, 1994 meeting in Seattle, WA:*

This was my first POSIX meeting. Given this fact, my perspective on the proceedings of the meetings attended should be viewed with a large pinch of salt. This report is not necessarily factually accurate or binding in any way.

The POSIX.1 working group (WG) was small, with four seniors and two freshmen (including me) present the first day. Both of us freshmen were accepted warmly, so there were two less freshmen on the last day of the meeting. The WG was now growing – proof of life and success!

The Seattle WG meeting was dominated by outstanding interpretations and ballot issues on Draft 11 of the POSIX.1a ballot. While other issues were discussed, the ballot status and issues surrounding Checkpoint/Restart seemed to keep resurfacing. Issues discussed during the week included:

- Trailing Slashes
- Application conformance
- Optional environment functions like *walkenv*(), *savenv*(), and *loadenv*()
- Checkpoint/Restart
- Removable Media Study Group

### Trailing slashes

The problem with the current standard is lack of clarity in the definition of pathname in the case that the pathname contains one or more trailing slashes.

It was decided that this is really a problem with the definition of "pathname resolution," and not with trailing slashes in particular. Thus the solution lay in clarifying the definition of "pathname resolution."

A proposal was put forth to tighten the requirement on pathnames ending in trailing slashes to allow them only in the case of existing directories – the behavior of BSD based systems. The definition of pathname resolution will now include something like the following:

> "A pathname that ends with one or more trailing slashes shall be interpreted as if it ended with slash-dot, except that the slash-dot does not count in {PATH_MAX} calculations."

### Application conformance

Two points were proposed here. First, the need for an additional definition to be added to POSIX.1 from C language – a definition that would allow for an application to conform to both POSIX.1 and to ISO 9899 C. Second, a plea for consistency in the wording of definitions in the different POSIX standards – at least those that relate to conformance issues.

The additional definition was that of a "Rigorously Conforming POSIX.1 C Application." This is defined as an application that requires the services and facilities only described in ISO/IEC 9945-1 and ISO/IEC 9899. Furthermore, with regard to the use of the C language, it shall not produce output dependent on any unspecified, undefined, or implementation-defined behavior, and shall not exceed any minimum implementation limits, as defined in ISO 9899.

The second point was about consistency in wording referred at least in part to the difference in the definitions of Strictly Conforming POSIX.1 and POSIX.2 Applications.

The language above was taken from the "Application conformance issues" paper by Derek Jones of Knowledge Software. This is not a complete definition of the "Rigorously Conforming POSIX.1 C Application." The complete definition can be got directly from the paper.

### Optional environment functions

There was a discussion of the environmental functions (*xxxenv*()) in POSIX.1a. The WG felt that such functions were probably useful in threaded environments, and should thus probably be optional. There was some discussion about *walkenv*() actually making things worse for threaded environments. The final decision was to discuss at the next meeting if the functions should be optional and if there could be an alternative function to *walkenv*() that would be thread-safe. [*Subsequent telephone meetings of the WG have decided to remove these contentious functions from the next draft of POSIX.1a. Ed.*]

### Checkpoint/Restart

There was a joint meeting with the Fault Tolerant (SRASS) group and the Super Computing Profile (POSIX.10) groups about checkpoint/restart. There are still a lot of conflicting goals for this. The POSIX.1 WG would like to see a useful minimal interface specification, something that can be implemented and used by a reasonable set of applications. The SRASS group would like to see the name changed from checkpoint/restart to something like "batch services," since the functions do not go far enough for their particular needs, and were invented originally within the batch profile.

The POSIX.1 WG would like to see a common name agreeable to all groups. It will oppose the possibility of two similar but different interfaces coming into existence for checkpoint/restart.

### Removable Media Study Group

This group discussed issues associated with tapes (generally true for any removable media) and how work supporting it might proceed. They expect to submit a Project Authorization Request (PAR) in January describing the scope and goals of the group.

The next meeting will be in sunny Ft. Lauderdale FL, January 15-21, 1995. I enjoyed the last meeting enough to be planning on going to Fort Lauderdale (well, it is Florida in January).

## Report on POSIX.4: Realtime Interfaces

*Joe Gwinn <GWINN@sud2.ed.ray.com> reports on the October 17-21, 1994 meeting in Seattle, WA:*

### Why are pthreads taking so long?

As many of you may have heard, final approval of the pthreads draft POSIX standard (was called P1003.4a, now renamed P1003.1c by the IEEE) was delayed for six months by a negative coordination ballot from the POSIX.5 Working Group (the Ada folk). The ensuing dispute finally bubbled up to the SEC (Sponsor Executive Committee, the governing body of POSIX), who at the POSIX.5 Working Group's request stalled POSIX.1c and sent the two Working Groups (POSIX.5 and POSIX.4) off to come to a fuller understanding of each other's position and issues, in the hope that the problems would then be resolved.

After three months of weekly telephone conferences and much email, the issues were indeed clarified, but the core differences remained irreconcilable. At the next SEC meeting, in July 1994, the POSIX.5 WG asked for further delay and also that the P1003.1c Balloting Group be queried by letter, while the POSIX.4 WG asked that the stall be rescinded and the matter closed. The SEC accepted POSIX.5's request for a query, but voted to limit the stall to 3 more months, to allow time for the query responses to be received.

The results of the survey of the pthreads balloting group was unequivocal. If the change requested by the POSIX.5 WG was incorporated into P1003.1c, approval would drop from 82% to something like 60%, way below the required 75%. If the change was not incorporated, approval would drop by less than one percent. A total of 186 balloters responded yes or no to the query. To state the result baldly, if the changes requested by the Ada folk were accepted, pthreads would be dead, and we would have to start over.

At the October 1994 POSIX meeting, the SEC voted to allow P1003.1c to resume progress towards standardization, and to allow the POSIX.5 WG's objections to accompany it.

(Because standards require consensus, not unanimity, the majority of standards have unresolved objections.) The next step is CD Registration and concurrent balloting at ISO, followed by presentation to the IEEE Standards Board. This will take some months, but further changes to P1003.1c (now at draft 10) are likely to be minimal to nonexistent, and the Ada logjam is broken.

Why was Ada allowed to hold pthreads up? IEEE standards require consensus, not just majority. This is necessary if the resulting standards are to be widely implemented and used. However, it does permit a vocal and determined minority to delay things.

By now, many readers are wondering just what was this core difference that caused so much trouble. As one might expect from the history of POSIX, signals were to blame. Specifically, according to P1003.1c, in a multi-threaded process, *sigprocmask*() is not required to (but is allowed to) mask signals globally, for all threads within the process. The change requested by the POSIX.5 WG was to require (rather than just allow) *sigprocmask*() to globally mask and unmask signals to all threads of the process. In either case, this would be in addition to the per- thread masking controlled by *pthread_sigmask*(). The Technical Editor and Reviewers of P1003.1c demurred, saying that requiring *sigprocmask*() to have global effect would lead to timing races in multi-threaded processes, particularly when those processes reside on multiple processors, and that having two masks could lead to use-update conflicts, and suggested that the requested change would also cause Ada implementations some trouble. The POSIX.5 WG disagreed. It was suggested that the POSIX.5 WG simply require implementors of P1003.5b, the draft Ada binding to Realtime POSIX (1003.1b-1993), to support the global *sigprocmask*() option. The POSIX.5 WG declined, saying that they feared that the UNIX platform vendors would not in fact support the option just to support Ada, leaving Ada high and dry.

There were many other arguments, too many to recount here. The root problem seems to be that the fundamental model of Ada, a language plus an operating system, differs greatly from the model of C and UNIX, and the Ada folk are having some difficulty coming to terms with having lost the language wars to C, and thus being forced to bend to what they consider to be an inferior model. Even within the US Department of Defense, Ada's star is waning.

## What else is the POSIX 1003.4 Working Group doing?

The POSIX.4 WG has 3 other major projects: P1003.4b (now renumbered as P1003.1d), P1003.1i, and "P1003.4d," discussed below.

POSIX.4b (P1003.1d), 130 pages, currently in ballot resolution, contains a number of realtime interfaces and options that arrived too late to be included in 1003.1b-1993 (which itself consists of POSIX.1 and POSIX.4 combined). The major new interfaces and options are:

- *spawn*(), a functional merger of *fork*() and *exec*(), needed both for efficiency and for allowing use on platforms lacking memory management hardware;
- a sporadic-server scheduling policy, used to prevent asynchronous high-priority processing from totally consuming the computer;
- cpu-time clocks and timers, used to both measure and bound use of cpu by processes;
- *devctl*(), the successor to the *ioctl*() of classic UNIX;
- Interrupt Control, a set of interfaces intended to allow direct application-level control of devices such as array processors and radar signal processors; and
- Advisory Information, a set of interfaces that allow an application to declare to the kernel that, for instance, a specified file will be read sequentially, allowing the kernel to optimize performance.

A number of existing interfaces are also being augmented by the addition of variants supporting time-outs.

P1003.1i (also known as "POSIX.4 technical corrections") consists of correcting the POSIX.4 interfaces to remedy clashes detected when POSIX.4 was merged into POSIX.1-1990 to yield 1003.1b-1993, as well as minor problems discovered by early implementors of 1003.1b-1993. P1003.1i is a fast-track effort with a very limited and specific scope, and is expected to go to ballot about January 1995. As the changes are small, simple, and few, approval is expected to be rapid.

"POSIX.4d" (no formal IEEE number assigned as yet), approved as a project by the SEC at the October 1994 POSIX meeting, and for which a 73-page draft already exists, contains a number of realtime interfaces and options that arrived too late to be included in POSIX.1d. The major new interfaces and options are:

- Typed Memory, a set of interfaces supporting the *mmap*()-like mapping of diverse kinds of physical memory (e.g., SRAM, DRAM, ROM, EPROM, EEPROM) via multiple and/or diverse physical paths used for instance to access special hardware and memory via attached VME busses;

- *nanosleep_abs*(), a high-resolution *sleep*() allowing the user to specify when to awaken, rather than how long to sleep;

- Barrier Synchronization, a set of interfaces intended to support efficient implementation of parallel DO/FOR loops on massively parallel computers;

- Reader/Writer Locks, used to allow efficient parallel access to data in situations where reads vastly outnumber writes;

- Spinlocks, a very fast synchronization primitive for use on shared-memory multiprocessors; and

- Persistent Notification for Message Queues, an option for POSIX.1b-1993 Message Queues.

POSIX.13 draft 6, 100 pages, in ballot resolution and at about 75% approval, is a family of four related realtime profiles ranging in size from the very small through a full-featured platform conforming to essentially all of POSIX.1b-1993 (POSIX.1 plus POSIX.4) and POSIX.1c (threads), with realtime options chosen. The smaller profiles specify just that subset of POSIX interfaces needed to "clothe" widely-used small kernels such as pSOS, WindRiver, and VRTX32, which although very similar in function differ greatly in interface details. Standardization of these interfaces will yield the same benefits for embedded and realtime as standardization of UNIX did for workstations. In addition, the POSIX.13 interfaces are chosen to allow multi-computer distributed systems to be built, such as those used in factory automation. Such systems are typically set up as a hierarchy, with a few large-profile machines at the top, and a large number of smaller profile machines at the bottom controlling this or that piece of machinery, perhaps with an intermediate layer of supervisory machines between top and base, and all communicating with peers, superiors, and subordinates as needed.

### Why are POSIX Document and Working Group Names so Confusing?

Good question; answer lost in prehistory. Sorry. The WGs are currently named (well, numbered) after the first standard they produced. When the WGs later took on added work, the WGs retained their original names. To further the confusion, the IEEE renamed all the documents and standards, and even the POSIX faithful are having some difficulty keeping the names straight. For example, the POSIX.4 Working Group was founded to produce the POSIX.4-1993 standard, known as "P1003.4" in its days as a draft standard. The POSIX.4 WG later started the POSIX.4a, POSIX.4b, and POSIX.13 draft standards; all but POSIX.13 have now been renamed. There is now a push to change WG names to an approximation of English, while the standards will continue to be numbered. The current set of non-numerical names aren't noticeably more informative than the numbers they replace. Stay tuned.

## A View from the Chair

*Lowell Johnson <3lgj@unirsvl.rsvl.unisys.com> reports on A View From the Chair:*

The Portable Applications Standards Committee, PASC, is obviously going through a period of change, and nobody can be sure exactly what the future will bring, but a couple of things are fairly certain.

The first is that we will have to create quality standards with the reduced participation level we have experienced recently. Even though the economy will probably be improving, I do not think we will see a significant increase in participation for a variety of reasons. Companies are dissatisfied with how long it takes us to generate a standard and many are not even sure they want a complete set of open system standards (they might have to comply with them all).

So the second certain change will be that we have to develop or improve our processes so that we can reduce the time between project initiation and the granting of final ISO standard status. There are several things we will be discussing about how we can accomplish this, but I know from personal experience that huge standards, especially those over 1000 pages, take unacceptably long to both write and ballot.

A more subtle change will be in the types of standards we produce. The recent decision by JTC1 to allow public specifications to enter the fast track process to become ISO standards (under certain conditions) will mean that popular subjects, developed or supported by broad user bases, may go that route to standardization, rather than go through a formal standards development organization (like IEEE).

I think we will still have some major items to work on, but the emphasis will change in some situations. We will have to provide more small standards on specialized topics and to fill holes not covered by existing major standards. Also, someone will have to define environments where all the base standards that have been developed can work to form an integrated whole. Fortunately, we know how to do that: they are called profiles.

You all know how difficult it is to define an open system, since basically everyone has their own definition, and most of them are not truly open (at least in the sense of portability). While contemplating this (and trying to write another article), I began thinking that the concept of a "Strictly Open System" would be more useful.

A Strictly Open System would strictly conform to all its underlying standards, and nothing else. Strict conformance to a base standard means conforming to all the interfaces defined in it, but with no extensions. Therefore, a Strictly Open System would not rely on any extensions to any standard.

Unfortunately, we all know that no system is strictly conforming since we do not have a complete set of system standards, and even if we did, most vendors use specific (and often undefined) hooks into the system to increase performance or otherwise improve salability. There is also the problem of how such a system could be proved to be conforming.

One possibility (although very gross) would be to measure the amount of documentation it would take to specify all the extensions used. However, I do not think even this is practical for most systems, but it would make an interesting thought experiment to estimate how large this pile of hypothetical documentation would be. Obviously, a perfect Strictly Open System would not require any of this documentation since no extensions would be needed.

A simple view of our future would be to work on anything which could be used to reduce this imaginary pile of extensions documentation. We will work on many other things (such as profiles), and may work in some new areas (such as application conformance), but we will also have to develop new ways to work with non-traditional standards groups.

# USENIX Supports Student Participation and Professional Development

The USENIX Association sponsors an Educational Outreach Program for faculty members of computer science departments around the world. These faculty liaisons provide their students access to USENIX publications, upcoming events as well as conference scholarships for full-time students. If you would like more information on the benefits or are interested in becoming a liaison please contact Diane DeMartini via email *<diane@usenix.org>*.

The following student benefits are offered:

- A $500 cash prize is awarded for the Best Student Paper at the annual Winter Technical Conference. (Students are eligible also for Best Paper and other awards.)

- Membership in USENIX for full-time Students is only $25. As a member, your benefits include:

- Free subscription to *;login:*, technical features, system administration tips and techniques, international calendar of events, SAGE News, media reviews, Snitch Reports from the USENIX representative and others on various ANSI, IEEE, and ISO standards efforts, and much more

- Free subscription to *Computing Systems*, the refereed technical quarterly published with The MIT Press

- Discounts on registration fees for the large, multi-topic technical conference, the System Administration conference (LISA), and the various symposia addressing single topics such as object-oriented technologies, security, operating systems, high-speed networking, and mobile computing – as many as seven technical meetings every year

- Discounts on proceedings from USENIX conferences and symposia and other technical publications. Discounts on the USENIX Association book published by The MIT Press. Now available: *The Evolution of C++: Language Design in the Marketplace of Ideas*, edited by Jim Waldo of Sun Microsystems Laboratories

- Discounts on BSDI, Inc. products. BSDI information: 800 800-4BSD

- Discounts on the five volume set of 4.4 BSD manuals plus CD published by O'Reilly & Associates and USENIX

- Savings (10-20%) on selected publications from McGraw-Hill, The MIT Press, Prentice Hall, John Wiley & Sons, O'Reilly and Associates, and UniForum

- Special subscription rates to *The LINUX Journal*. Phone: 206 527-3385

- Opportunity to join SAGE, the System Administrators Guild

- And maybe most important, participation in leadership in the systems community

For membership information please contact:

The USENIX Association  
2560 Ninth Street, Suite 215  
Berkeley, CA 94710  
510/528-8649  
FAX 510/548-5738  
Email: *<info@usenix.org>*  
WWW URL: *<http://www.usenix.org>*

## BOOK REVIEWS

# The Bookworm

*by Peter H. Salus*
*<peter@uunet.uu.net>*

Happy New Year!

I've got a real complaint this month: someone at Wiley has been infected by some sort of virus. I got one book that's printed in blue, making the illustrations invisible and the volume unreadable and another (Gilster on Mosaic) that's in brown. Dark inks were real good, troops. It meant that one could actually read the text. Modish, but silly, folks.

I want to discuss (or at least mention) about a dozen books this month, so here goes.

### TCP/IP

The big book is here. Several years ago, I remarked that I had thought that Rich Stevens' *Advanced UNIX Programming* was a tabletop when it arrived. Well, now he's created another "brick." 1200 pages. It's the second volume of his *TCP/IP Illustrated.* And this time he's gotten help from Gary Wright. I try not to lie, so I'll confess that even though I've had several months with the volume, I've not read all of it. But I have looked at stuff in it. And I've not found it lacking. Wright and Stevens have gone through the 20K lines of TCP/IP code in the 4.4BSD release. In fact, they've used the code from 4.4-Lite, so you can follow the protocols at work license free and actually gain understanding. This and its predecessor make for a very different pair of books than the Comer/Stevens quartet. If you want to really understand the way the protocol suite works (as opposed to merely using it), you're going to need these two books.

### PGP

*Pretty Good Privacy* is an encryption program. PGP 2.6 and beyond can be used freely for non-commercial purposes in the US. In Europe and Asia, don't worry about it. Written by Phil Zimmermann, the release of PGP was precipitated by the 1991 Omnibus Crime Bill (S. 266), which provided that "providers of electronic communications services . . . permit the government to obtain the plain text contents of communications . . ." (This is the same clause that gave rise to the Clipper Chip fiasco.) In July 1991, the language was removed from the bill, but Zimmermann read the language as declaring PGP illegal. Anyway, PGP is RSA public key cryptography for the masses. And Simson Garfinkel has produced an appropriately pretty good book about it, one which looks at privacy and cryptography, not merely PGP. I especially liked the Appendices B, C and D, which explain the installation of PGP on UNIX, DOS and Macintosh platforms. As produced by ORA, this is a pretty good volume.

### Mosaic

Whenever I see the word "Mosaic," I have a frisson that the book will be about Judaic law or a tobacco virus or the art of Ravenna. These days, it's "none of the above." The books I get with Mosaic in the title appear to be about Tim Berners-Lee's graphical interface.

I found *The Mosaic Handbook for the X Window System* disappointing. I'm used to ORA books that punch out the hard-core data to me. More than a third of this one has nothing to do with Mosaic at all: its a general sketch of the Internet, an

Books reviewed in this column

Gary R. Wright & W. Richard Stevens, **TCP/IP Illustrated, Volume 2: The Implementation**. Addison-Wesley, 1995. 1200pp. ISBN 0-201-63354-X. $52.75.

Simson Garfinkel, **PGP: Pretty Good Privacy**. O'Reilly & Associates, 1994. 400pp. ISBN 1-56592-098-8. $24.95.

Dale Dougherty, Richard Koman & Paula Ferguson, **The Mosaic Handbook for the X Window System**. O'Reilly & Associates, 1994. 288pp. + CD-ROM. ISBN 1-56592-095-3. $29.95.

Paul Gilster, **The Mosaic Navigator**. John Wiley & Sons, 1994. 256pp. ISBN 0-471-11336-0. $16.95.

John S. Quarterman & Smoot Carl-Mitchell, **The E-Mail Companion**. Addison-Wesley, 1994. 318pp. ISBN 0-201-40658-6. $19.95.

Dave Taylor, **Teach Yourself UNIX in a Week**. SAMS Publishing, 1994. 714pp. ISBN 0-672-30464-3. $28.

Rosalind Resnick & Dave Taylor, **The Internet Business Guide**. SAMS Publishing, 1994. 418pp. ISBN 0-672-30530-5. $25.

Andrew S. Tanenbaum, **Distributed Operating Systems**. Prentice Hall, 1995. 614pp. ISBN 0-13-219908-4. $57.

Peter G. Neumann, **Computer-Related Risks**. ACM Press/ Addison-Wesley, 1995. 307pp. ISBN 0-201-55805-X. $24.75.

Samuel P. Harbison & Guy L. Steele Jr., **C A Reference Manual**. 4th edition. Prentice Hall, 1995. 455pp. ISBN 0-13-326224-3.

Evi Nemeth, Garth Snyder, Scott Seabass, & Trent R. Hein, **UNIX System Administration Handbook**. 2nd edition. Prentice Hall, 1995. 750+pp. + CD-ROM. ISBN 0-13-151051-7. $48.

Robert King Ables, **The Keys to Successful UNIX System Management**. Prentice Hall, 1995. 316pp. ISBN 0-13-948381-0.

exposition of WWW, and one on GNN. Only when you get to Chapters 5-7 will you hit the real stuff. It's well-presented. The CD-ROM is effective. But I found the volume relatively low-key, too much into the mass market. (The Windows version comes with two diskettes; the Mac version with one.)

On the other hand, Paul Gilster's *The Mosaic Navigator* is aimed at the mass market and does a really good job. I mentioned two of Gilster's other books in my last column. I keep his *Finding it on the Internet* next to my table. *The Mosaic Navigator* is, effectively, a massive expansion of the section on Mosaic in *Finding it* . . . . This is a good piece of work at a reasonable price. It lacks the CD-ROM, but it's $13 cheaper than *The Mosaic Handbook*, too.

### Email

Quarterman and Carl-Mitchell have written a volume on email that should enable readers of this column to set up their parents (or grandparents) in confidence. In 300 pages of non-obfuscatory prose and examples, John and Smoot have produced an easy, yet not condescending, handbook that will enable uninitiates to communicate effectively. Topics like "how electronic mail works," "mail addressing and routing," "mailing lists," and "anonymous FTP by mail" are handled very well. John and Smoot have managed to escape the 200-300 pages of lists of lists and news groups. No bodies, no blood. Check it out!

### Beginners' books

I usually avoid real beginners' books, because they annoy me. Dave Taylor's *Teach yourself UNIX* was recommended to me as an exception. It is. While it's tough to work yourself through the lessons when you know the answers, it looks to me as though Taylor has fulfilled his promises. This may be the only book that stands up to ORA's *Learning* volume, which has been my favorite in three editions for nearly a decade. But ORA's book is intended to be worked-through in a few hours. I really believe that beginners can learn the system from Taylor in a week. But it won't be an easy week!

I'm afraid that I can't generate quite the same enthusiasm for Resnick and Taylor. There is some good information here and there, but most of the volume is made up of fluff. On the other hand, as I recently demonstrated use of telnet and FTP to a multi-millionaire real estate mogul, maybe this is indeed the right level. Certainly, if we are really at 30 million Matrix users and Vint Cerf's estimate of 400 million by 2000 is right, there will be an awful lot of tourists on the Infobahn who need this sort of work.

### Distribution rights

In 1987, Andy Tanenbaum's *Operating Systems* was a big hit (recall that it came with the card you could send in for the MINIX code). *Distributed Operating Systems* now completes Tanenbaum's task. This is a very good book. Tanenbaum (like the Coulouris volume I discussed last year) uses the case study approach: Amoeba, Mach, Chorus, and DCE are the cases here; Coulouris, et al. had Clouds, but not DCE. Tanenbaum's choice of DCE is an interesting one, as DCE isn't really an OS, but a tool that builds a distributed environment on top of other systems. If I were teaching a course, this would be a tough choice. I think I'd come down on the side of Coulouris, though. But it's close.

### Risky business

Peter Neumann has written a book which entertained me on several consecutive air trips. Taking tales and data from the risks forum, Neumann has put together a volume that no one concerned with computing can afford to miss: from disaster stories to financial fraud to increasing security, it's all here. Highly recommended.

### Repeat performances

Harbison and Steele's *C A Reference Manual* is now over ten years old. The new fourth edition is bigger and better. If you've been using 2nd ed. (like me) or 1st or 3rd, you need a new one.

Nemeth, et al. is six years old. The new edition is bigger, revised and absolutely necessary for anyone involved in UNIX system administration. It now comes with a CD-ROM. It has a new foreword by Allman and McKusick to go with the old one (1988) by Ritchie. Indispensable.

[I got R.K. Ables *The Keys to Successful UNIX System Management* in the same mail with Nemeth, et al. There's just no comparison. But my comprehension is taxed by the fact that they're from the same publisher!]

## The FDDI Handbook: High-Speed Networking Using Fiber and Other Media

By Raj Jain (Addison-Wesley, 1994, ISBN: 0-201-56376-2) 528 pages.

*Reviewed by George V. Neville-Neil <gnn@netcom.com>*

FDDI has been called everything from the next step in networking to a technological abomination. What FDDI really stands for is Fiber Distributed Data Interface. First proposed in the early 80's by an ANSI Standards Committee (ASC X3T9.5), it has since been issued as a formal standard, built, and deployed. There is now a follow-on standard, FDDI-II which tries to address issues that the original standard did not, including multimedia data transport.

The FDDI Handbook provides a comprehensive resource on FDDI technology, starting from the hardware level and working its way up through the protocol layers. It's an excellent executive summary listing the pros and cons of FDDI as well as a good first introduction to the technology. This chapter is especially suited to non-technical people. I call it the "how to get your boss to buy FDDI" chapter.

Following the overview the discussion begins with the lowest protocol layer, Media Access Control. This layer is what defines FDDI and how nodes coordinate access to the network. Next is a discussion of the Medium Independent Physical Layer (aka PHY), the layer closest to the hardware not directly tied to the type of hardware you are using.

The book then takes a break from the mechanics of the protocol to discuss the fundamentals of optical communication. This serves as a general overview (for those of you who skipped physics) of how optical fibers, lasers, and LEDs can be used for communication. This section is particularly good in its treatment of this subject: I was neither bored, nor insulted by the explanations.

Having covered the basics of optical communication the book moves on to the different types of physical components that can be used to make up an FDDI network. The chapters are: "Optical Components: Physical Layer Medium Dependent;" "Single-Mode Fiber: SMF-PMD)" "Low-Cost Fiber (LCF-PMD);" "FDDI on Twisted-Pair Copper Cables;" and "FDDI on SONET." Each chapter talks in depth about the costs and benefits of using a particular type of hardware to build the network.

Moving up from the physical layer, the book covers a series of management layers, beginning with station management (SMT). Stations are the basic entities on an FDDI network, such as hosts, bridges, routers, or concentrators. The station management layer must initialize and control the network, perform status monitoring, and isolate and recover from faults, in order to keep the network running. The connection and ring management (CMT and RMT) chapter talks about the layers that allow communication between the nodes on the network, as well as the management of the ring topology itself. The last chapter in this group describes the way higher level network management software sees FDDI using the SNMP protocol.

The next chapters describe extensions to the basic FDDI protocol. FDDI-II is meant to address voice network concerns (like those of the phone company). It is especially tuned to carrying normal telephone traffic since many of the components are synchronized to an 8kHz clock, which is the sampling frequency of a voice phone call.

Logical Link Control describes how FDDI has adopted the IEEE's 802.2 LLC standards. LLC allows FDDI networks to be bridged to other non-FDDI networks such as Ethernet or Token Bus. The "TCP/IP and OSI Protocols on FDDI" chapter discusses implementing these higher level protocols.

The next several chapters are all about building an FDDI network for yourself – what I would call the "network administrator's friend." They are a set of how-to instructions so that you don't get burned when buying and installing equipment. The chapters include: "Buying and Installing Fiber Cables," "Cable Plant: Design and Analysis," "Buying FDDI Products: What to Look For."

The final chapters are all about keeping the network running once you have installed it: "Performance Under Heavy Load," "Performance Under Normal Load," "Analytical Models," "FDDI-II and Adapter Performance Issues," "Error Analysis," and "Conformance Testing."

After reading this book, I find that it is useful as much more than just a handbook for FDDI. I think it would also be an excellent textbook for a course on networking. Its style is easy to read. It has numerous illustrations and side-bars that cover some of the more obscure ideas in a fun and interesting way. Each chapter has a summary, bibliography, and a set of self-test exercises that also make it a natural for teaching.

As a handbook it is excellent. After reading any one chapter, I felt that I really understood the issues that were addressed by that part of the protocol, and also I gained some insight into what the designers had thought about when writing the standard. The organization of this book is excellent as well. I have already recommended it to our local systems administrator because of the several chapters on planning, installing, and maintaining FDDI. I really cannot speak highly enough of this book, and I commend the author on a job well done.

## Japanese Information Processing

by Ken Lunde, (O'Reilly & Associates, Inc., 1993, ISBN: 1-56592-043-0) 470 pages, $29.95.

*Reviewed by Jeff Haemer*
*<jsh@canary.com>*

Few Americans, when asked to recite the alphabet, can get all the way to the end without breaking into song. You know: A, B, C, D, E, F, G, H, I, J, K, L-M-N-O-P, . . . What do you suppose the Japanese sing instead? Horse, monkey, door, cow, Rain-shower, East, toaster-oven, . . .

Ken Lunde's new book, *Japanese Information Processing*, doesn't answer that question but, if you're interested in the subject, you should buy it anyway.

The biggest issue in dealing with Japanese, and many other Asian languages, is that they don't use an alphabet. Because ideographs – symbols for whole words – are central to Japanese writing, the size of the character set is both large and open to negotiation. Essentially all of the book is devoted to this issue and its logical consequences. The chapter titles make this clear:

- Overview of Japanese Information Processing
- The Japanese Writing System
- Japanese Character Set Standards
- Japanese Encoding Methods
- Japanese Input
- Japanese Output
- Japanese Information Processing Techniques
- Japanese Text Processing Tools
- Using Japanese E-mail and News

The target audience isn't experienced programmers – early chapters explain ideas like "bytes" and "characters." I never felt that the book was talking down to me. I really wish I'd had this book the first time I'd tried to make a piece of software handle a large character set; and even though I've now "NLSthetized" (say it fast enough and it sounds like "anesthetized") several largish applications, the book still gave me useful information.

Quick: What's the distinction between JIS X 0208-1983 and JIS X 0208-1990? What shift sequence indicates the JIS data stream is shifting to one-byte ASCII characters? What's the distinction between Kyoiku Kanji and Gakushu Kanji? You could keep the answers in your head, but this book lets you look them up. I read the book, cover-to-cover but that's probably atypical. This is a heavily detailed reference book; you should expect your copy to become dog-eared and splattered with coffee stains. Nearly half the book is appendices, and even in the main text there are tons of tables, exhibits, and figures.

The figures are particularly interesting. First, Ken Lunde works for Adobe Systems, so the samples of printed Japanese are beautiful. (On the other hand, I will warn you that if you write in books, as I do, the paper's so thin that your ink will bleed through to the other side.)

Second, all multi-byte encoding schemes, of which the Japanese have several (there is no consensus counterpart to ASCII), really assign characters to locations in a multi-dimensional array. Accordingly, the author has dozens of pictures of code sets showing what parts of each character set occupy what portion of the square or cube of all possible two- or three-byte combinations. Candidly, I haven't found these pictures useful yet, but I suspect that that's me, not the charts. I tend to find descriptions in words and code work best for me; the book has those, too.

The code, which is confined to Chapter 7, is available via anonymous ftp. This is becoming typical with O'Reilly offerings, and we should applaud this heartily, not take it for granted. A few months ago, I was reading a fine book from a different publisher and got the urge to try out some of the code in it. On investigation, I discovered that the publisher would sell me software that would run the code under UNIX for only $150 (. . . but the DOS version is just $50).

I will say that the code looks odd to me. Here's a sample fragment:

```
while ((pl = fgetc(in)) != EOF){
                     /* Read one byte into pl
                     until end-of-file is
                     detected */
       if (pl == ESC) {
                     /* If escape character
                     is detected */
            temp = fgetc(in);
                     /* Read next byte to
                     temp */
```

Every line has a comment, and there are far more *else*s than I'm used to seeing. The easiest description is that it looks like C written by an assembly-language programmer who's been writing Pascal. That quibble aside, the code looks sane. Everything that I scrutinized closely seemed correct.

One small warning. There are a few topics missing that you might expect to see in a book with this title. One missing topic is any discussion of how to write or modify applications to handle Japanese data. There is only a glancing mention of locales, no discussion of `wchar_t` and friends, and no mention of the impact of really large character sets on algorithm design. This is not a book on internationalization, it's the book you need as background for dealing with Japanese data.

Some cultural issues are also omitted. The text does a fine job of summarizing some, including word-wrap, hyphenation, white-space, and vertical text, but doesn't discuss Imperial dates or numeric formats.

A third topic that the text does little but mention is the Japanese variants of EBCDIC, which use a shift-out/shift-in encoding scheme. I lead with this forbidden topic to break gently to the potential reader that this isn't a UNIX-specific book. Chapter 8, which covers text-processing software, has much more coverage of Macintosh software than of any other kind.

Mind you, if I were the author I might have omitted these topics, too; still, I'm the reviewer so I should mention omissions, if only as a heads-up to the folks who may buy the book if I say that it's really good. And it's really good.

While on the topic of really good books about Japanese characters, I'll also recommend *Remembering the Hiragana* and its companion *Remembering the Katakana.* [James W. Heisig, *Remembering the Hiragana, a complete course on how to teach yourself the Japanese syllabary in 3 hours,* (Japan Publications, 1987) $6.00; Helmut Morsbach, Kazue Kurebayashi, and James W. Heisig, *Remembering the Katakana, with a supplement on Learning How to Remember,* (Japan Publications, 1990) $6.00.] Each book is less than a hundred pages long and will teach you one of the Japanese syllabaries in under three hours, spread over a week or so. "Huh? What's a 'syllabary'?" you may be asking yourself. Unlike the Chinese, who write everything with ideographs, the Japanese also have a phonetic script – two of them, actually, katakana and hiragana – which they use to write things like particles, endings, and foreign words. For example, in the word "selling," "sell" might be written with an ideograph and "ing" written in hiragana.

A syllabary is a phonetic script that has one symbol for each syllable. In contrast, writing systems with one symbol for each sound are called "alphabets." In other words, in a syllabary, "ing" would be a single symbol. (Japanese can get away with a syllabary because there are only about fifty distinct syllables in Japanese. This is because nearly all Japanese syllables are a consonant followed by a vowel, which gives Japanese its characteristic sound: To-yo-ta, Su-ba-ru, Fu-ji-ya-ma, and so on.)

You can learn the Japanese phonetic scripts, then, by memorizing about a hundred symbols. That's fewer than the tens of thousands needed to learn the Kanji, but it's still a barrier for many. These books tunnel through that barrier. I'll quote from Heisig's introduction to the first book:

> A visiting professor [. . .] casually tossed the challenge at my feet one evening over a mug of beer: "Why hasn't anybody figured out an easy way to learn the syllabary?" I didn't know if anyone had or not, but the next morning I took a sheet of white paper and wrote in large bold letters the title you see on the cover of this book. I set the paper on the corner of my desk and resolved not to write another line until I was satisfied I had grounds to justify its boast. If this sounds like the kind of guy you'd like, you'll like the books, too.

## Computer-Related Risks

by Peter G. Neumann (ACM Press/Addison-Wesley, 1995, ISBN 0-201-55805-X.) 307pages, $24.75.

*Reviewed by Rob Kolstad*
*<kolstad@bsdi.com>*

I read this book after reading *RISKS-DIGEST* for a while and kind of expected a compendium the various columns, maybe sorted.

I almost recoiled at the intensive criticism of various large engineering systems (e.g., space shuttles) in Chapter 2. I felt that Monday morning quarterbacking was too easy in the various efforts to succeed at enormously complex endeavors. I pressed on, hoping that I was overreacting.

Happily, the succeeding chapters were more and more interesting. Once finished with transportation problems, the books moves through all sorts of various vulnerabilities and weaknesses that large (and small) systems have demonstrated through the years. These anecdotes and their analyses would make fine text for any engineer trying to anticipate the kinds of uses and abuses that a product might see over its life span.

By the time I came upon the final segments with "prescriptions" in them, I was with the author all the way. I believe he is right-on with most of his ideas and commentaries on ways to improve software (and, generally, large system) engineering.

The book runs to 351 pages and also has a sixteen page index. It is quite readable and easy to swallow in small chunks (a typical scenario takes far less than a minute to read). Other than the intro and final prescription chapters, the book need not be read in order.

I enjoyed the book except for a bit of the second chapter – but I might have been a bit stressed about that one since I spend so much time working in software systems that can see the kinds of problems it described.

It's an easy read with lessons for all levels of software engineers and system administrators.

# USENIX/SAGE Tutorials at UniForum

**March 12 - 13, 1995**
**Dallas, Texas**

## Tutorials

Mark your calendar. If you could not attend LISA or the USENIX Technical Conference in New Orleans, USENIX is offering two days of tutorials as part of UniForum's pre-conference program on Sunday and Monday, March 12 and 13 in Dallas. The topics are sure to interest you and are listed below.

The UniForum '95 conference will take place on Tuesday through Thursday, March 14-16. There are many tracks to choose from that be useful to you.

### Sunday, March 12, 1995

#### Internet System Administrator's Tutorial

*Ed DeHart, Computer Emergency Response Team*

Intended Audience: Users and system administrators of UNIX systems, especially system administrators of UNIX systems connected to a wide area network based on TCP/IP such as the Internet. Some system administrator experience is assumed.

#### UNIX Power Tools - Getting the Most Out of UNIX

*Rob Kolstad, Berkeley Software Design, Inc.*

Intended Audience: Programmers, managers, and system administrators wanting to learn more about the powerful development tools available on UNIX. This tutorial emphasizes software development rather than system administration.

### Monday, March 13, 1995

#### Firewalls - Achieving Security in an Internet Environment

*Tina Darmohray, Lead, UNIX System Administration Team, Lawrence Livermore National Laboratories; Rob Kolstad, Berkeley Software Design, Inc.*

Intended Audience: System administrators, programmers, technical and operational managers, and all interested professionals involved in securing computer networks and/or internetwork gateways. Pre-requisites are a knowledge of TCP/IP, DNS, and sendmail.

#### The Law and the Internet

*Daniel Appelman, Attorney, Heller, Ehrman, White & McAuliffe*

Intended Audience: Anyone interested in the legal issues arising out of the increasing use and popularity of the Internet, in particular system administrators, contract administrators and company executives who need to develop policies about doing business electronically.

## Registration Information

For more information, call the Interface Group at 617 449 5554, and they will send you a conference brochure containing the full program. If you just need a registration form, call 617 449 5554 to register by fax. From a touch-tone phone, enter code 79 and key in your fax number. A registration form will be faxed to you. Just complete it and fax it back.

Registration information is also available on the World Wide Web. The URL is: *http://www.uniforum.org*.

# Call for Participation Tcl/Tk Workshop 95

The third annual Tcl/Tk workshop, co-sponsored by Unisys, Inc. and the USENIX Association will be held this summer, July 6-8, 1995 in Toronto, Ontario, Canada. The workshop has a number of aims:

- To act as a focus for Tcl/Tk research.
- To provide a mechanism for the announcement and publication of Tcl/Tk research.
- To promote active collaboration amongst the Tcl/Tk community.
- To provide a framework for discussing open issues and possible solutions.

It is intended that the program shall be as general as possible and non-exclusive. The form of the workshop shall consist of:

- Invited addresses by distinguished Tcl/Tk researchers
- Oral presentations
- Formal demonstrations similar to the oral presentations in format
- Informal demonstrations for commercial applications

## Paper/Demonstration Submission Information

Papers and demonstrations reporting original research are sought on any aspect of Tcl/Tk. Researchers are invited to submit papers describing their work with Tcl/Tk. The papers may cover, but are not limited to:

- Extensions (system, widgets, etc.)
- Systems using Tcl/Tk
- Experience building systems using Tcl/Tk
- Proposals for new directions

Submitted papers may be a maximum of ten pages.

There will be two demonstration formats. Formal demonstrations will be similar to a paper in format, but will be longer (approximately 30 minutes), and will consist of a live demonstration of software. This format is intended for non-commercial research. The conference will provide a non-networked UNIX machine running Tcl/Tk for these demonstrations. Informal demonstrations will be held in a separate place and time (probably during the evening), and are intended to include commercial product demos, etc. The conference will not supply any hardware for these demonstrations. Submitted demonstrations should include:

- A statement of which demonstration format (formal or informal) you are submitting to, and why a demonstration is the best way to present the work
- A statement about the commercial status of the technology
- A statement of who the presenter is (developer, designer, marketer, etc.)
- A detailed description of the necessary equipment you will need for the demonstration
- A two page summary of the work which will be published in the proceedings. This summary must be in the format described at the end of this Call For Participation

Papers and demonstrations will be reviewed by the program committee. The committee will not accept papers that have been submitted for publication elsewhere. If accepted, authors of papers will be invited to present their work in an approximately 20 minute time slot. Demonstrations will be given approximately 30 minutes. Full versions of the written papers and two page summaries of the demonstrations will be printed in the proceedings to be published by the USENIX Association.

Original papers and demonstrations must be submitted by 5:00pm, March 3, 1995. Submissions must be in English, and no fax submissions will be accepted. You must submit one paper copy and one electronic copy via email *<tclpapers@ usenix.org>*.

Use 8.5x11 inch or A4 paper for the paper copy, and postscript or text for the electronic copy. Make sure each copy of the packet is stapled, not loose or held by clips. Be sure to include an email address so we can acknowledge receipt of your submission. Be sure to allow enough time for your submission to arrive by the indicated deadline.

You may submit a VHS NTSC format video to support your paper. This may be appropriate for certain papers or demonstrations which are particularly interactive or visual in nature. If you choose to do this, you must also submit 5 (FIVE) copies of the videotape. Note that videos are for review purposes only, and will not be published. It will not be possible to return the videos. Upon acceptance of a paper or demonstration submission, you will receive a Guidelines for Authors Kit that will include formatting instructions, an author's checklist, and consent form.

NOTE: Even if you submit a paper or demonstration to the conference, you must also register for the conference. See registration information below.

### Send paper and demonstration submissions to:

USENIX Association
2560 Ninth Street, Suite 215
Berkeley, California, 94710 USA
(510) 528-8649
(510) 548-5738 (FAX)
Email: *<tclpapers@usenix.org>*

### Important Dates:

| | |
|---|---|
| Deadline for papers and demos: | 5:00pm, March 3, 1995 |
| Acceptance of papers and demos: | April 14, 1995 |
| Camera-ready and electronic versions due: | May 23, 1995 |

### Registration Information: July 6th - 8th 1995

Put these dates in your diary now. The workshop is open to the entire Tcl/Tk community. Registration for the workshop will cost US$250. This includes a copy of the proceedings, lunches, coffee, snacks and a reception/dinner.

Our experience has shown that an exchange of ideas works best when there is a small to moderate number of participants. For this reason, we are interested in bringing together active users of Tcl/Tk. Because of this and space constraints, we will be limiting participation in this workshop to 150 people. If you would like to attend the workshop, please submit a short outline (no more than 1/2 a page) describing your reason for attending the workshop. This request should also include the following information:

First and Last Name
Company or Institution
Mailing address
Telephone number
Internet address
Any special dietary requirements
Any special needs (e.g. handicap)

NOTE: Submitting a paper does not automatically register you for the conference. You must submit this information even if you make a submission to the conference. Upon acceptance for attending the workshop, you will receive instructions for payment as well as information for car rental and GST rebate. Registration can be submitted electronically by sending email to *<w95@system9.unisys.com>* or by mailing/faxing to:

Tcl/Tk Workshop 95
c/o Unisys Canada Inc.
61 Humidified Road
Scarborough, Ontario, M1S 5A9 Canada
(416) 297-2520 (FAX)

The workshop is being held at the Royal York hotel. A special room rate has been given to the workshop of 119 Canadian dollars, plus taxes. For this rate, reservations must be made before June 1, 1995 and you should reference the Tcl/Tk workshop. The hotel address is:

Royal York Hotel
100 Front Street West
Toronto, Ontario, M5J 1E3 Canada
800 441-1414 (inside North America)
416 368-2511 (elsewhere)

The workshop is sponsored by Unisys Inc. and USENIX Association.

### Co-chairs:

Ben Bederson, *University of New Mexico*
*<bederson@cs.unm.edu>*
Will Wilbrink, *Unisys Inc.*
*<will@system9.unisys.com>*

### Program Committee:

John Ousterhout, *Sun Microsystems, Inc.*
Steve Uhler, *DSP Group*
Gerald Lester, *Computerized Processes Unlimited*
Charley Crowley, *University of New Mexico*
David Richardson, *University of Michigan*
Wayne Christopher, *ICEM CFD Engineering*
Kevin Kenny, *General Electric Corporate R&D*
Brian Smith, *Cornell University*

To get complete tutorial descriptions, refer to postings on *comp.org.usenix*, access the World Wide Web, URL: *http://www.usenix.org* or send mail to our automatic mailserver at: *<info@usenix.org>*.

# 5th USENIX UNIX Security Symposium

JUNE 5–7, 1995
MARRIOTT HOTEL
SALT LAKE CITY, UTAH

## IMPORTANT DATES

**DATES FOR REFEREED PAPER SUBMISSIONS:**

- ♦ Extended abstracts due: February 13, 1995
- ♦ Program Committee decisions made: March 8, 1995
- ♦ Camera-ready final papers due: May 1, 1995
- ♦ Registration materials available: March 1995

USENIX®

The UNIX® and Advanced
Computing Systems
Professional and Technical
Association.

**ANNOUNCEMENT AND PRELIMINARY CALL FOR PAPERS**

## SPONSORED BY THE USENIX ASSOCIATION, IN COOPERATION WITH: THE COMPUTER EMERGENCY RESPONSE TEAM (CERT), IFIP WG 11.4, AND UNIFORUM

### OVERVIEW

The goal of this symposium is to bring together security practitioners, researchers, system administrators, systems programmers, and others with an interest in computer security as it relates to networks and the UNIX operating system.

This will be a three day, single-track symposium consisting of tutorials, refereed and invited technical presentations, and panel sessions. The first day will be devoted to tutorial presentations. Two days of technical sessions will follow the tutorials.

### TUTORIALS

♦ **June 5**

This one-day tutorial program is designed to address the needs of both technical and management attendees. The tutorials will supply overviews of various security mechanisms and policies. Each will provide specifics to the system and site administrator for implementing numerous local and network security precautions, firewalls, and monitoring systems.

### KEYNOTE AND TECHNICAL SESSIONS

♦ **June 6–7**

The keynote address by Stephen T. Walker, founder and president of Trusted Information Systems, will begin the technical sessions program. Mr. Walker will speak on information security and privacy in computing. Mr. Walker is an electronics engineer and computer systems analyst with over 25 years of experience in system design and program management; particularly extensive is his experience with the design and implementation of large scale computer networks and information systems. He is nationally recognized for his pioneering work on the DoD Computer Security Initiative, the establishment of the National Computer Security Center, and the formation of the Defense Data Network. He is a member of the Computer System Security and Privacy Advisory Board, established by the Computer Security Act of 1987.

The technical sessions program, in addition to presentations of refereed papers, will include invited talks, and possibly panel sessions. There will also be two evenings available for Birds-of-a-Feather sessions (BoFs) and Works-in-Progress Reports (WiPs). The program committee invites you to submit proposals, ideas, or suggestions for these presentations; your suggestions may be submitted to the program chair via email to *security@usenix.org* or by post to the address given on the following page.

The program committee will formally review and accept papers for presentation at the symposium and publish them in the symposium proceedings. USENIX will provide provide the proceedings free to technical session attendees; additional copies will be available for purchase from USENIX.

## Symposium Topics

Presentations are being solicited in areas including but not limited to:

- User/system authentication
- File system security
- Network security
- Security and system management
- Security-enhanced versions of the UNIX operating system
- Security tools
- Security incident investigation and response
- Computer misuse and anomaly detection

## Refereed Paper Submissions

Submissions must be received by February 13, 1995. Full papers should be 10–15 pages. Instead of a full paper, authors may submit an extended abstract which discusses key ideas. Extended abstracts should be 5–7 pages long (about 2500–3500 words), not counting references and figures. The body of the extended abstract should be in complete paragraphs. The object of an extended abstract is to convince the reviewers that a good paper and presentation will result. All submissions will be judged on originality, relevance, and correctness.

An individual program committee member will be assigned to shepherd each accepted submission through preparation of the final paper. The assigned member will act as a conduit for feedback from the committee to the authors. Camera-ready final papers are due May 1, 1995.

Please accompany each submission by a cover letter stating the paper title and authors along with the name of the person who will act as the contact to the program committee. Please include a surface mail address, daytime and evening phone number, and, if available, an email address and fax number for the contact person.

If you would like to receive detailed guidelines for submission and examples of extended abstracts, you may telephone the USENIX Association office at +1 510 528 8649, or email to *securityauthors@usenix.org* or to the program chair.

The USENIX UNIX Security conference, like most conferences and journals, requires that papers not be submitted simultaneously to another conference or publication and that submitted papers not be previously or subsequently published elsewhere. Papers accompanied by "non-disclosure agreement" forms are not acceptable and will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

## Where To Submit

Please send one copy of a full paper or an extended abstract to the program committee via one of the following methods. All submissions will be acknowledged.

- Preferred method: email (PostScript or ASCII) to *securitypapers@usenix.org*
- Alternate method: postal delivery to Fred Avolio, Trusted Information Systems, 3060 Washington Road, Glenwood, MD 21738
- Phone: +1 301 854 6889
- Fax: +1 301 854 5363

## Program Committee

- **Program Chair:**
  Fred Avolio, *Trusted Information Systems, Inc.*

Steve Bellovin, *AT&T Bell Laboratories*

Bill Cheswick, *AT&T Bell Laboratories*

Ed DeHart, *CERT*

Ed Gould, *Digital Equipment Corporation*

Marcus Ranum, *Trusted Information Systems, Inc.*

Jeff Schiller, *MIT*

Gene Spafford, *COAST Laboratory, Purdue University*

## For Registration Information

Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available beginning in April 1995. If you wish to receive the registration materials, please contact USENIX at:

- USENIX Conference Office
  22672 Lambert Street
  Suite 613
  Lake Forest, CA 92630
  USA

Phone: +1 714 588 8649
Fax: +1 714 588 9706
Email: *conference@usenix.org*

For more information about USENIX and its events, access the USENIX Resource Center on the World Wide Web. The URL is *http:// www.usenix.org*. Or send email to our mailserver at: *info@usenix.org*. Your message should contain the line: *send catalog*. A catalog will be returned to you.

**ANNOUNCEMENT & PRELIMINARY CALL FOR PAPERS**

# USENIX Conference on Object-Oriented Technologies (COOTS)

JUNE 26–29, 1995
MONTEREY, CALIFORNIA

## Dates For Refereed Paper Submissions

- Submissions due: March 6, 1995
- Notification to authors: April 3, 1995
- Camera-ready final papers due: May 15, 1995

## Preliminary Program Committee

- Program Chair: Vincent F. Russo, *Purdue University*
- Tutorial Program Chair: Doug Lea, *SUNY Oswego*

Mark Linton, *Silicon Graphics, Inc.*

Chris Pettus, *Taligent*

Jim Waldo, *SUN Microsystems Labs.*

Murthy Devarokonda, *IBM Watson Research Labs*

Ted Goldstein, *SUN Microsystems Labs.*

Paul Leach, *Microsoft*

Luca Cardelli, *Digital Systems Research Center*

The COOTS conference is designed to be a showplace for advanced development work in object-oriented technologies. The conference will emphasize research and experience derived from efforts to use object-oriented techniques to builds complex systems that meet real world needs.

The COOTS conference will begin with two days of tutorials. The tutorial program will offer a selection of tutorials from among several tracks. We expect tutorial topics to include:

- distributed object systems (CORBA, etc.)
- object-oriented network programming
- alternative object-oriented languages
- advanced techniques in memory management
- efficient and effective class design

If you are interested in offering a tutorial, proposals are due December 1, to Doug Lea, tutorial program chair (***dl@g.oswego.edu***).

Two days of technical sessions will follow the tutorials. Proceedings of the conference will be published by USENIX and will be provided free to technical session attendees; additional copies will be available for purchase from USENIX.

Like the USENIX C++ Conferences and Advanced Topics Workshops from which it is derived, COOTS will emphasize the advanced engineering aspects of object technology. While papers covering work in C++ are encouraged, the conference is broader in scope than its ancestors and invites submissions describing results and work in other object-oriented or object-based languages.

## Conference Topics

We seek papers describing original work concerning the design, implementation, and use of object-oriented technologies. Questions regarding a topic's relevance may be addressed to the program chair via electronic mail to ***russo@cs.purdue.edu***.

Potential topics include:

- work on object-oriented programming languages (C++, Modula-3, Eiffel, etc.)
- implementations of commercial object infrastructures (CORBA, NextStep, OLE-II, SOM/DSOM, etc.)
- interface description languages
- distributed object systems
- unique applications of and experiences with object-oriented technologies

## Refereed Paper Submissions

Submissions must be received by March 6, 1995. Full papers should be 10 to 15 pages. Instead of a full paper, authors may submit an extended abstract which discusses key ideas. Extended abstracts should be 5–7 pages long (about 2500–3500 words), not counting references and figures. The body of the extended abstract should be complete paragraphs. The object of an extended abstract is to convince the reviewers that a good paper and presentation will result. While, by acceptance of extended abstracts, we intend to stimulate industrial participation, submission of extended abstracts by academics is in no way discouraged.

All submissions will be judged on originality, relevance, and correctness. Each accepted submission will be assigned a member of the program committee to act as its shepherd through the preparation of the final paper. The assigned member will act as a conduit for feedback from the committee to the authors. Camera-ready final papers are due May 15, 1995.

Please accompany each submission with a cover letter stating the paper title and authors along with the name of the person who will act as the contact to the program committee. Please include a surface mail address, daytime and evening phone number, and, if available, an email address and fax number for the contact person.

If you would like to receive detailed guidelines for submission and examples of extended abstracts, you may telephone the USENIX Association office at +1-510-528-8649, or email to ***cootsauthors@usenix.org*** or to the program committee chair.

The COOTS conference, like most conferences and journals, requires that papers not be submitted simultaneously to another conference or publication and that submitted papers not be previously or subsequently published elsewhere. Papers accompanied by "non-disclosure agreement" forms are not acceptable and will be returned to the author(s) unread. All submissions are held in the highest confidentiality prior to publication in the Proceedings, both as a matter of policy and in accord with the U.S. Copyright Act of 1976.

## Where To Submit

Please send one copy of a full paper or an extended abstract to the program committee via one of the following methods. All submissions will be acknowledged.

- ♦ Preferred Method: email (Postscript or ASCII) to ***cootspapers@usenix.org***
- ♦ Alternate Method: postal delivery to
  USENIX COOTS Conference
  c/o Dr. Vincent F. Russo
  Department of Computer Sciences
  Purdue University
  West Lafayette, IN 47907 USA
  Telephone: +1 317 494 6008

## Dates For Refereed Paper Submissions

- ♦ Submissions due: March 6, 1995
- ♦ Notification to authors: April 3, 1995
- ♦ Camera-ready final papers due: May 15, 1995

## For Registration Information

Materials containing all details of the technical and tutorial programs, registration fees and forms, and hotel information will be available beginning in April 1995. If you wish to receive the registration materials, please contact USENIX at:

- ♦ USENIX Conference Office
  22672 Lambert Street
  Suite 613
  Lake Forest, CA 92630
  USA
- ♦ Telephone: +1 714 588 8649
- ♦ Fax: +1 714 588 9706
- ♦ Internet: ***conference@usenix.org***

For more information about USENIX and its events, access the USENIX Resource Center on the World Wide Web. The URL is ***http://www.usenix.org***. Or send email to our mailserver at: ***info@usenix.org***. Your message should contain the line: ***send catalog***. A catalog will be returned to you.

USENIX®

The UNIX and Advanced Computing Systems Professional and Technical Association

# 9th Systems Administration Conference (LISA '95)

**SEPT. 18–22, 1995**
MARRIOTT HOTEL
MONTEREY, CALIFORNIA

## ANNOUNCEMENT & CALL FOR PARTICIPATION

The 9th Systems Administration (LISA) Conference, sponsored by USENIX and SAGE, is widely recognized as the leading technical conference for system administrators. Historically, LISA stood for "Large Installation Systems Administration," back in the days when having a large installation meant having over 100 users, over 100 systems, or over one giga-byte of disk storage. Today, the scope of the LISA conference includes topics of interest to system administrators from sites of all sizes and kinds. What the conference attendees have in common is an interest in solving problems that cannot be dealt with simply by scaling up well-understood solutions appropriate to a single machine or a small number of workstations on a LAN.

The theme for this year's conference is "New Challenges," which includes such emerging issues as integration of non-UNIX and proprietary systems and networking technologies, distributed information services, network voice and video teleconferencing, and managing very complex networks. We are particularly interested in technical papers that reflect hands-on experience, describe fully implemented and freely distributable solutions, and advance the state of the art of system administration as an engineering discipline.

## Tutorial Program

**Monday and Tuesday, September 18–19, 1995**
The two-day tutorial program offers up to five tracks of full and half-day tutorials. Tutorials offer expert instruction in areas of interest to system administrators of all levels, from novice through senior. Topics are expected to include networking, advanced system administration tools, Solaris and BSD administration, Perl programming, firewalls, NIS, DNS, Sendmail, and more.

To provide the best possible tutorial offerings, USENIX continually solicits proposals for new tutorials. If you are interested in presenting a tutorial at this or other USENIX conferences, please contact the tutorial coordinator, Daniel V. Klein:

♦ Phone: +1 412 421 0285; FAX: +1 412 421 2332; E-mail: *dvk@usenix.org*

## Technical Sessions

**Wednesday through Friday, September 20–22, 1995**
The three days of technical sessions consist of two parallel tracks. The first track is dedicated to presentations of refereed technical papers. The second track will accommodate invited talks, panels and Works-in-Progress (WIP) sessions.

## Conference Topics

Papers addressing the following topics are particularly timely; papers addressing other technical areas of general interest are equally welcome.

♦ Your plans for the year 2000
♦ Deployment of new networking technologies
♦ Coping with the commercialization of the internet
♦ Support models in use at your site
♦ Dealing with differences in UNIX implementations – migration and interoperability among BSD, SVR4, OSF and others
♦ Integration of UNIX-based with non-UNIX-based and proprietary systems and networking technologies (Mac, NT and DOS PCs)
♦ Application of emerging technologies (Mbone, Mosaic) to system administration
♦ Administration and security of distributed information services (WAIS, gopher, WWW) and network voice and video teleconferencing (Mbone)
♦ Experience supporting mobile and location-independent computing
♦ Experience with large (1000+ machine) networks, especially networks of SVR4-based systems
♦ Real-world experience with implementations of proposed system administration standards
♦ Unusual applications of commercial system administration software packages
♦ Application of operational planning techniques to system administration including measurements and metrics, continuous process improvement, automation, and increasing productivity
♦ File migration, archival storage & backup systems in extremely large environments
♦ Innovative tools and techniques that have worked for you

## Important Dates

**REFEREED PAPER SUBMISSIONS:**

| | |
|---|---|
| ♦ Extended abstracts due: | May 1, 1995 |
| ♦ Notification to authors: | June 5, 1995 |
| ♦ Final papers due: | August 1, 1995 |

**REGISTRATION MATERIALS AVAILABLE:**
♦ July, 1995

Co-sponsored by

USENIX®

The UNIX® and Advanced Computing Systems Professional and Technical Association

AND

SAGE
THE SYSTEM ADMINISTRATORS GUILD

- ♦ Managing high-demand and high-availability environments
- ♦ Migrating to new hardware and software technologies
- ♦ Administration of remote sites that have no technical experts
- ♦ Supporting MIS organizations on UNIX
- ♦ Real-world experiences with emerging procedural/ethical issues – e.g., developing site policies, tracking abusers, and implementing solutions to security problems
- ♦ Networking non-traditional sites (libraries, museums, K–12)

## Refereed Paper Submissions

An extended abstract is required for the paper selection process. Full papers are not acceptable at this stage; if you send a full paper, you must also include an extended abstract. "Extended" means 2–5 pages.

Include references to establish that you are familiar with related work, and, where possible, provide detailed performance data to establish that you have a working implementation or measurement tool.

Submissions will be judged on the quality of the written submission, and whether or not the work advances the state of the art of system administration. For more detailed author instructions and a sample extended abstract, send e-mail to: *lisa9authors@usenix.org* or call the USENIX office at +1 510 528 8649.

Note that LISA, like most conferences and journals, requires that papers not be submitted simultaneously to more than one conference or publication and that submitted papers not be previously or subsequently published elsewhere. Papers accompanied by "non-disclosure agreement" forms are not acceptable and will be returned unread. All submissions are held in the highest confidence prior to publication in the conference proceedings, both as a matter of policy and as protected by the U.S. Copyright Act of 1976.

Authors of an accepted paper must provide a final paper for publication in the conference proceedings. At least one author of each accepted paper presents the paper at the conference. Final papers are limited to 20 pages, including diagrams, figures and appendixes, and must be in troff, ASCII, or LaTeX format. We will supply you with instructions. Papers should include a brief description of the site, where appropriate.

Conference proceedings, containing all refereed papers and materials from the invited talks, will be distributed to attendees and will also be available from USENIX following the conference.

## Where to Send Submissions

Please submit extended abstracts for the refereed paper track by two of the following methods. All submissions will be acknowledged.

- ♦ E-mail to: *lisa9papers@usenix.org*; FAX to: +1 510 548 5738; Mail to: LISA 9 Conference, USENIX Association, 2560 Ninth Street, Suite 215, Berkeley CA USA 94710

To discuss potential submissions, and for inquiries regarding the content of the conference program, contact the program co-chairs at *lisa9chair@usenix.org* or at:

- ♦ Tina M. Darmohray, Lawrence Livermore National Laboratory, PO Box 808 L-510, Livermore CA USA 94550. +1 510 423 5999; FAX: +1 510 422 7869; E-mail: *tmd@usenix.org*
- ♦ Paul Evans, Synopsys, Inc., 700 East Middlefield Road, Mountain View CA USA 94043. +1 415 694 1855; FAX: +1 415 965 8637; E-mail: *ple@usenix.org*

## Invited Talks Track

If you have a topic of general interest to system administrators, but that is not suited for a traditional technical paper submission, please submit a proposal for a second track presentation to the invited talk (IT) coordinators at *itlisa@usenix.org* or to:

- ♦ Laura de Leon, Hewlett-Packard. +1 415 857 5605; FAX: +1 415 857 5686; E-mail: *deleon@hpl.hp.com*
- ♦ Peg Schafer, BBN. +1 617 873 2626; FAX: +1 617 873 4265; E-mail: *peg@bbn.com*

## Program Committee

♦ Program Co-chair:
Tina Darmohray, *Lawrence Livermore National Laboratory*
♦ Program Co-chair:
Paul Evans, *Synopsys, Inc.*
Paul Anderson, *University of Edinburgh*
Kim Carney, *Massachusetts Institute of Technology*
Rob Kolstad, *Berkeley Software Design, Inc.*
Bryan McDonald, *SRI International*
Marcus Ranum, *Trusted Information Systems, Inc.*
John Schimmel, *Silicon Graphics, Inc.*

## For Registration Information

All details of the technical and tutorial programs, registration fees and forms, and hotel information will be available in July, 1995. If you wish to receive the registration materials, please contact USENIX:

♦ USENIX Conference Office
22672 Lambert Street, Suite 613
Lake Forest, CA 92630 USA
Phone: +1 714 588 8649
Fax: +1 714 588 9706
E-mail: *conference@usenix.org*

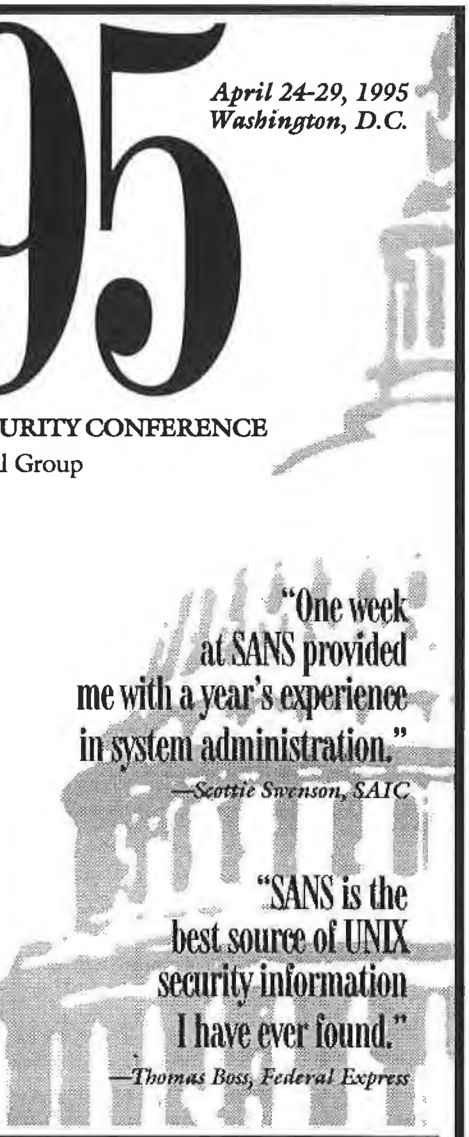
For more information about USENIX and its events, access the USENIX Resource Center on the World Wide Web. The URL is *http://www.usenix.org*. Or send e-mail to our mailserver at: *info@usenix.org*. Your message should contain the line: *send catalog*. A catalog will be returned to you.

## Vendor Display

**Wed. & Thurs., Sept. 20–21, 1995**

Well-informed vendor representatives will demonstrate products and services at the informal display. If your company would like to participate, please contact:

♦ Zanna Knight, +1 510 528 8649
FAX: +1 510 548 5738
E-mail: *display@usenix.org*



D582

# Prentice Hall PTR is pleased to recommend the following titles to USENIX members...

**USENIX members receive a 15% discount on orders**

____**UNIX System Administration Handbook, Second Edition,** Evi Nemeth/Garth Snyder, 0-13-151051-7
(15105-0) *List:* $48.00 *Members:* $40.80

____**Object-Oriented Modeling and Design,** James Rumbaugh, 0-13-629841-9
(62984-0) *List:* $54.00 *Members:* $45.90

____**Zen and the Art of the Internet, Third Edition,** Brendan Kehoe, 0-13-121492-6
(12149-1) *List:* $23.95 *Members:* $20.36

____**The Magic Garden Explained**, Bernard Goodheart/ James Cox, 0-13-098138-9
(09813-7) *List:* $38.00 *Members:* $32.30

____**Internetworking with TCP/IP, Vol. II Design, Implementation, and Internals,** Douglas E. Comer/ David L. Stevens, 0-13-472242-6
(47224-1) *List:* $61.33 *Members:* $52.13

____**SCO Performance Tuning Handbook,** Gina Miscovich/David Simons, 0-13-102690-9
(10269-9) *List:* $42.00 *Members:* $35.70

____**Object-Oriented Programming**, Peter Coad/ Jill Nicola, 0-13-032616-X
(03261-5) *List:* $48.00 *Members:* $40.80

____**Internetworking with TCP/IP, Vol. III Client Server Programming and Applications for the BSD Socket Version,** Douglas E. Comer and David L. Stevens, 0-13-474222-2
(47422-1) *List:* $53.00 *Members:* $45.05

____**Internetworking with TCP/IP, Vol. III Client Server Programming and Applications for the AT&T TLI Version**, Douglas E. Comer and David L. Stevens, 0-13-474230-3
(47423-9) *List:* $53.00 *Members:* $45.05

____**The Internet Message: Closing the Book with Electronic Mail,** Marshall T. Rose, 0-13-092941-7
(09294-0) *List:* $50.00 *Members:* $42.50

____**The Standard C Library**, PJ Plauger, 0-13-131509-9
(13150-8) *List:* $37.80 *Members:* $32.13

____**All About Administering the NIS+, Second Edition** Rick Ramsey, 0-13-309576-2
(30957-5) *List:* $42.00 *Members:* $35.70

____**The Simple Book: An Introduction to Internet Management** Marshall T. Rose, 0-13-177254-6
(17725-3) *List:* $55.00 *Members:* $46.75

____**Networking Operations on UNIX SVR4,** Mike Padavano, 0-13-613555-2
(61355-4) *List:* $50.00 *Members:* $42.50

____**Solaris Porting Guide,** Sunsoft ISV Engineering 0-13-030396-8
(03039-5) *List:* $52.00 *Members:* $44.20

____**Multiprocessor System Architectures,** Ben Catanzaro 0-13-089137-1
(08913-6) *List:* $44.00 *Members:* $37.40

____**The HP-UX System Administrator's "How To" Book** Marty Poniatowski, 0-13-099821-4
(09982-0) *List:* $32.00 *Members:* $27.20

____**UNIX System V Performance Management,** Edited by Phyllis Eve Bregman and Sally A. Browning 0-13-016429-1
(01642-8) *List:* $29.95 *Members:* $25.45

____**SCO® UNIX® Operating System System Administrator's Guide,** Santa Cruz Operation, 0-13-012568-7
(01256-7) *List:* $39.00 *Members:* $33.15

## HERE'S HOW TO ORDER:

**CALL**
**800-880-6818**

OR WRITE:

**CompuBooks**
**Route 1, Box 271-D,**
**Cedar Creek, TX 78612**

OR INTERNET:

**70007.1333@CompuServe.com**
**(GO CBK on CompuServe)**

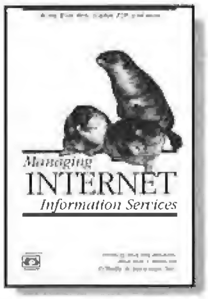
**WE SHIP ANYWHERE!**

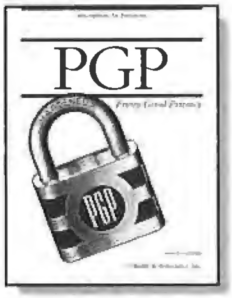
**FOR MORE INFORMATION, OR QUANTITY ORDERS, PLEASE CALL 201-592-2657**

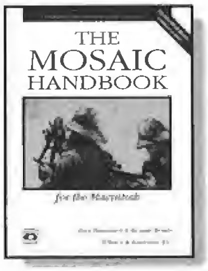
# Commemorating 25 years of UNIX

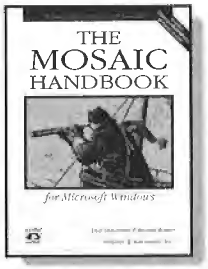
This year UNIX marks its 25th anniversary. We think its something to celebrate. After all, we owe our livelihoods to UNIX. Over the years it's given us plenty to explore, learn, teach, and talk about. It's enabled us to do good work—work that's filled with adventure, fun, and creative challenge. We couldn't ask for much more than that. Except, of course, for a little cake and ice cream.

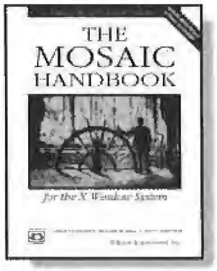
You might like to celebrate with these new titles from O'Reilly: *Managing Internet Information Services*, *The Mosaic Handbook* (with software for three platforms, including Microsoft Windows, X, and the Macintosh), *X User's Tools*, *Motif Tools*, *PGP: Pretty Good Privacy*, and *Multi-Platform Code Management.* Call or write to us at the numbers below for details. Mention code ALOG.





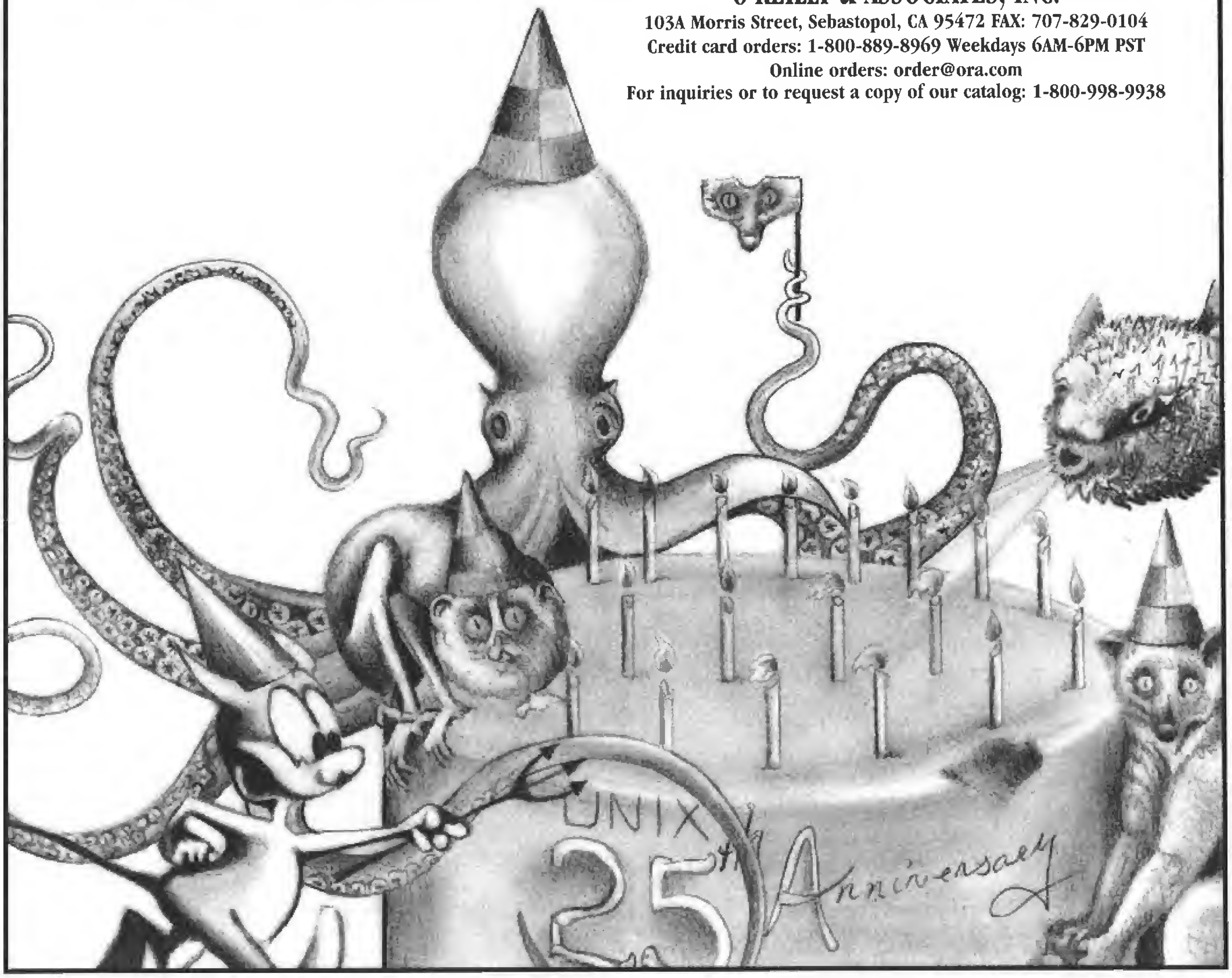


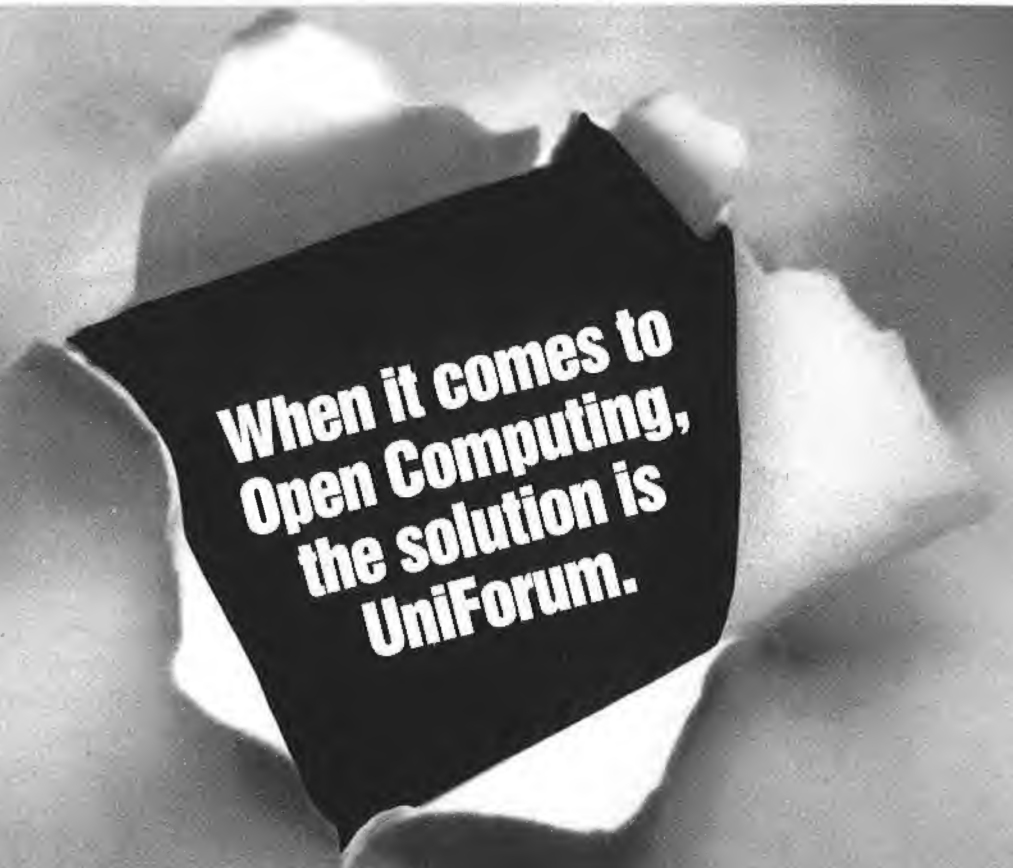




**O'REILLY & ASSOCIATES, INC.**

**103A Morris Street, Sebastopol, CA 95472 FAX: 707-829-0104**
**Credit card orders: 1-800-889-8969 Weekdays 6AM-6PM PST**
**Online orders: order@ora.com**
**For inquiries or to request a copy of our catalog: 1-800-998-9938**





# UniForum'95

The Official Conference and Exposition for Open Computing Solutions

**March 14–16, 1995**
**Dallas Convention Center • Dallas, Texas**

## REGISTER BY FAX!

Call 617-449-5554, enter Code 70 and have your fax number ready — we'll fax back your registration form within 24 hours!

**NEW!**

**UNIFORUM ON-LINE...for the latest information!**

**Internet**
World Wide Web URL: http://www.uniforum.org

**Now managed by The Interface Group, the producer of COMDEX.**
**Sponsored by UniForum, The International Association of Open Systems Professionals.**

## Just Published by UniForum

The MOSES Whitepapers
The Collected Whitepapers from the Massive Open Systems Environment Standard
UniForum

# The MOSES Whitepapers

## The Collected Whitepapers from the Massive Open Systems Environment Standard

For the first time in one volume the collected Whitepapers from *MOSES: The Massive Open Systems Environment Standard* is now available from UniForum.

The MOSES Project is a collective effort by large open systems shops to define the operational production environment needed to ensure the highest levels of system availability and reliability. The Whitepapers are aimed primarily at MIS professionals who operate large data centers, especially those in UNIX environments. The contributors are all experts who wrote the selections as part of their efforts to improve their own shops and to create a dialog on the best practices for perfecting the UNIX data center. The Whitepapers, while written by and for UNIX operations, are largely applicable to any operating system or database.

*The MOSES Whitepapers* are arranged as 12 stand-alone chapters that can be used individually or as a collection to provide data center operators with the guidelines they need to perform their duties more efficiently.

## The Whitepapers include:

- **General Introduction**
  John C. Black, US WEST NewVector Group, Inc.
- **Service Level Support Agreements**
  Bill Holt, US WEST NewVector Group, Inc.
- **Event Monitoring and Alarming**
  Steve Starck, US WEST NewVector Group, Inc.
- **Backup and Recovery Requirements**
  Mike Farnham, Burlington Coat Factory
- **Change Management**
  Michelle L. Trout, Sequent Computer Systems
- **Benchmark Standards/Metrics and Capacity Planning**
  Paul McKenney, Sequent Computer Systems
- **Data Archiving Requirements**
  Ron Hawkins, Millipore
- **Job Scheduling**
  Dan Poulin, Millipore
- **Interface and Infrastructure Requirements**
  Bill Hughes, Sequent Computer Systems
- **Security**
  Barry Harness, British Telecom
- **Disaster Planning/Recovery**
  Scott McMahan, Oracle Corporation
- **UNIX System Administration Hiring Process**
  Bill Holt, US WEST NewVector Group, Inc.

*All General Members of UniForum will receive a copy of The MOSES Whitepapers as part of their membership benefits.*

**Copies of *The MOSES Whitepapers* are available to all UniForum General and Trial Members at substantial discounts:**

| QUANTITY | MEMBER PRICE | TRIAL MEMBER PRICE |
|---|---|---|
| 1-10 copies | $10.00 | $25.00 |
| 11-49 copies | $ 8.00 | $20.00 |
| 50 copies or more | $ 7.00 | $17.00 |

*Plus shipping and handling*

| Shipping & Handling Charges | US/Canada/Mexico | International |
|---|---|---|
| If your mechandise total is... | Add... | Add... |
| UP to $25.00 | $5.00 | $7.50 |
| $25.01 to 50.00 | 7.00 | 9.50 |
| $50.01 to 75.00 | 9.00 | 11.50 |
| $75.01 or more | 11.00 | 13.50 |

To order, please call UniForum today at **1-800-255-5620**, or **(408) 986-8840**. Have your Visa, MasterCard, Discover or American Express Card ready for fastest order processing. Orders may be sent to UniForum. Please send checks or money orders (US dollars only; California residents please add applicable sales tax), to:

**UniForum**
**MOSES Whitepapers**
**2901 Tasman Drive, Suite 205**
**Santa Clara, CA 95054-1100**

**Credit card orders may also be faxed to us at (408) 986-1645**

UniForum™
The International Association of Open Systems Professionals

# What is Open Systems and where can you find the information you need to keep up in today's competitive open systems environment?

## Find out by joining UniForum: The International Association of Open Systems Professionals.

### The Advantages of Membership:

Your annual membership is a resource treasure chest. Among the many valuable benefits you'll receive are:

- *UniForum Monthly*: The Magazine for Open Systems Professionals
- *UniNews* - the authoritative voice of UniForum - now on-line
- *The Open Systems Products Directory* - comprehensive, accurate, indispensable
- Reduced fees at all UniForum conferences and seminars
- Technical and Standards publications: timely, reliable and useful
- Discounts on many outstanding industry publications
- Discounts on leading commercial hardware and software products
- Discounts on courses offered by leading training organizations
- Discounts on exclusive new shareware selections
- Services such as discounts on car rentals, travel and mail list rentals
- Eligibility to serve on UniForum committees and the Board of Directors

## UNIFORUM MEMBERSHIP APPLICATION FORM *(please print)*

Name ______________________________ MR/MS

Title ______________________________

Company ______________________________

Address ______________________________

Mail Stop ______________________________

City ____________ State ____________ Zip ____________

Country ____________ Telephone (______) ____________

E-mail (where we will send UniNews) ____________ Fax (______) ____________

### General Membership** $125

Includes *UniForum Monthly* magazine, *UniNews* electronic newsletter, *Open Systems Products Directory,* technical publications, recruitment advertising service, and discounts on: UniForum conference registration and shareware CD-ROMs, purchases of software and hardware, educational seminars and special classes, additional UniForum publications and other industry publications, computer books, CD-ROMs and training services.

** Overseas postage:
air $100 or surface $60 (no additional postage necessary for Canada, Mexico or Puerto Rico) $________

TOTAL AMOUNT ENCLOSED $________

### Method of Payment:

Select one: ❑ Check payable to UniForum * ❑ Money Order In US Dollars
❑ MasterCard ❑ Visa ❑ American Express

Credit card number ____________ Expiration date ____________

Signature ______________________________

*Payments must be included in U.S. dollars. All checks must be drawn on a U.S. bank. Credit card orders can be accepted by telephone.
**Overseas surface delivery takes up to six weeks.

**Send application and payment to: UniForum, 2901 Tasman Drive, Suite 205, Santa Clara, CA 95054-1100**
**Tel: (408) 986-8840 (800) 255-5620 Fax: (408) 986-1645**

Prices subject to change without notice. Contact UniForum for discounts and shipping and handling charges on bulk orders
UniForum is a registered trademark of UniForum. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company Limited

UniForum.
The International Association of Open Systems Professionals

## There's Never Been a Better Time for an Outstanding Value

**As the unstoppable move to open systems gathers even more momentum you need a strong and independent association that can help you achieve your professional goals.**
**An association that can magnify your voice, which can educate, inform and inspire.**
**You need and deserve an association that provides value for your money. In short you need UniForum.**
**With a value this good, the real question might be, "Can you afford to be without it?"**
**Join today!**

ORDER FORM

USENIX

## CONFERENCE & WORKSHOP PROCEEDINGS

| *Qty* | *Proceedings* | | *Member Price* | *Non-Member• Price* | *Subtotal* | *Overseas Postage* | *Total* |
|---|---|---|---|---|---|---|---|
| | WINTER/SUMMER CONFERENCES | | | | | | |
| ____ | New Orleans | Winter '95 | 30 | 39 | $________ | 20 | $________ |
| ____ | Boston | Summer '94 | 25 | 33 | $________ | 18 | $________ |
| ____ | San Francisco | Winter '94 | 30 | 39 | $________ | 20 | $________ |
| ____ | Cincinnati | Summer '93 | 25 | 33 | $________ | 18 | $________ |
| ____ | San Diego | Winter '93 | 33 | 40 | $________ | 25 | $________ |
| ____ | San Antonio | Summer '92 | 23 | 30 | $________ | 14 | $________ |
| ____ | San Francisco | Winter '92 | 30 | 39 | $________ | 22 | $________ |
| ____ | Nashville | Summer '91 | 32 | 38 | $________ | 22 | $________ |
| ____ | Dallas | Winter '91 | 28 | 34 | $________ | 18 | $________ |
| ____ | Anaheim | Summer '90 | 22 | 22 | $________ | 15 | $________ |
| ____ | Washington, DC | Winter '90 | 25 | 25 | $________ | 15 | $________ |
| ____ | Baltimore | Summer '89 | 20 | 20 | $________ | 15 | $________ |
| ____ | San Diego | Winter '89 | 30 | 30 | $________ | 20 | $________ |
| ____ | San Francisco | Summer '88 | 29 | 29 | $________ | 20 | $________ |
| ____ | Dallas | Winter '88 | 26 | 26 | $________ | 15 | $________ |
| ____ | Phoenix | Summer '87 | 35 | 35 | $________ | 20 | $________ |
| ____ | Washington, DC | Winter '87 | 10 | 10 | $________ | 15 | $________ |
| ____ | Atlanta | Summer '86 | 37 | 37 | $________ | 20 | $________ |
| ____ | Denver | Winter '86 | 25 | 25 | $________ | 15 | $________ |
| ____ | Portland | Summer '85 | 45 | 45 | $________ | 25 | $________ |
| ____ | Dallas | Winter '85 | 15 | 15 | $________ | 10 | $________ |
| ____ | Salt Lake City | Summer '84 | 29 | 29 | $________ | 20 | $________ |
| ____ | Washington, DC | Winter '84 | 25 | 25 | $________ | 15 | $________ |
| ____ | Toronto | Summer '83 | 32 | 32 | $________ | 20 | $________ |
| ____ | San Diego | Winter '83 | 28 | 28 | $________ | 15 | $________ |
| | LARGE INSTALLATION SYSTEMS ADMINISTRATION | | | | | | |
| ____ | LISA VIII | Sept. '94 | 22 | 29 | $________ | 10 | $________ |
| ____ | SANS III | April '94 | 15 | 20 | $________ | 9 | $________ |
| ____ | LISA VII | Nov. '93 | 25 | 33 | $________ | 12 | $________ |
| ____ | LISA VI | Oct. '92 | 23 | 30 | $________ | 12 | $________ |
| ____ | LISA V | Sept. '91 | 20 | 23 | $________ | 11 | $________ |
| ____ | LISA IV | Oct. '90 | 15 | 18 | $________ | 8 | $________ |
| ____ | LISA III | Sept. '89 | 13 | 13 | $________ | 9 | $________ |
| ____ | LISA II | Nov. '88 | 8 | 8 | $________ | 5 | $________ |
| ____ | LISA I | April '87 | 4 | 4 | $________ | 5 | $________ |
| | C++ | | | | | | |
| ____ | C++ Conference | April '94 | 24 | 28 | $________ | 20 | $________ |
| ____ | C++ Conference | Aug. '92 | 30 | 39 | $________ | 20 | $________ |
| ____ | C++ Conference | April '91 | 22 | 26 | $________ | 11 | $________ |
| ____ | C++ Conference | April '90 | 28 | 28 | $________ | 18 | $________ |
| ____ | C++ Conference | Oct. '88 | 30 | 30 | $________ | 20 | $________ |
| ____ | C++ Workshop | Nov. '87 | 30 | 30 | $________ | 20 | $________ |
| | SECURITY | | | | | | |
| ____ | UNIX Security IV | Oct. '93 | 15 | 20 | $________ | 9 | $________ |
| ____ | UNIX Security III | Sept. '92 | 30 | 39 | $________ | 11 | $________ |
| ____ | UNIX Security II | Aug. '90 | 13 | 16 | $________ | 8 | $________ |
| ____ | UNIX Security | Aug. '88 | 7 | 7 | $________ | 5 | $________ |
| | MACH | | | | | | |
| ____ | Mach Symposium III | April '93 | 30 | 39 | $________ | 18 | $________ |
| ____ | Mach Symposium | Nov. '91 | 24 | 28 | $________ | 14 | $________ |
| ____ | Mach Workshop | Oct. '90 | 17 | 20 | $________ | 9 | $________ |

| Qty | Proceedings | | Member Price | Non-Member Price | Subtotal | Overseas Postage | Total |
|---|---|---|---|---|---|---|---|
| | **DISTRIBUTED & MULTIPROCESSOR SYSTEMS (SEDMS)** | | | | | | |
| ____ | SEDMS IV | Sept. '93 | 24 | 32 | $______ | 14 | $______ |
| ____ | SEDMS III | Mar. '92 | 30 | 36 | $______ | 20 | $______ |
| ____ | SEDMS II | Mar. '91 | 30 | 36 | $______ | 20 | $______ |
| ____ | SEDMS | Oct. '89 | 30 | 30 | $______ | 20 | $______ |
| | **MICROKERNELS & OTHER KERNEL ARCH.** | | | | | | |
| ____ | Microkernels & Other Kernel... II | Sept. '93 | 15 | 20 | $______ | 9 | $______ |
| ____ | Microkernels & Other Kernel... I | Apr. '92 | 30 | 39 | $______ | 20 | $______ |
| | **GRAPHICS** | | | | | | |
| ____ | Graphics Workshop V | Nov. '89 | 18 | 18 | $______ | 10 | $______ |
| ____ | Graphics IV | Oct. '87 | 10 | 10 | $______ | 10 | $______ |
| ____ | Graphics III | Nov. '86 | 10 | 10 | $______ | 5 | $______ |
| ____ | Graphics II | Dec. '85 | 7 | 7 | $______ | 5 | $______ |
| | **OTHER WORKSHOPS/SYMPOSIA** | | | | | | |
| ____ | OS Design and Implementation | Nov. '94 | 20 | 27 | $______ | 11 | $______ |
| ____ | Very High Level Languages | Oct. '94 | 23 | 30 | $______ | 10 | $______ |
| ____ | High-Speed Networking | Aug. '94 | 15 | 20 | $______ | 9 | $______ |
| ____ | UNIX Applications Development | April '94 | 15 | 20 | $______ | 9 | $______ |
| ____ | Mobile Computing | Aug. '93 | 15 | 20 | $______ | 8 | $______ |
| ____ | File Systems | May '92 | 15 | 20 | $______ | 9 | $______ |
| ____ | UNIX Transaction Processing | May '89 | 12 | 12 | $______ | 8 | $______ |
| ____ | Software Management | Apr. '89 | 20 | 20 | $______ | 15 | $______ |
| ____ | UNIX & Supercomputers | Sept. '88 | 20 | 20 | $______ | 10 | $______ |
| | **SAGE Short Topics System Administration Series**<br>Short Topics System Administration Series | | | | | | |
| ____ | #1: Job Descriptions for System Administrators | | 5 | 7.50 | $______ | 3.50 | $______ |

*Discounts are available for bulk orders. Please inquire.*

**Total price of Proceedings** ________ **
**Calif. residents add sales tax** ________
**Total overseas postage** ________
**Total enclosed** ________

****If you are paying member price, please include member's name and/or membership number**______________________________

## PAYMENT OPTIONS*

____ Check enclosed - payable to USENIX Association   Account # ____________________ Exp.Date ________

____ Charge my: ____ VISA ____ MC   Signature ______________________________

____ Purchase order enclosed

* Outside the USA? Please make your payment in US currency by one of the following:
- Check - issued by a local branch of a US Bank
- Charge (VISA, MasterCard, or foreign equivalent)
- International postal money order

• If you are not a member and wish to receive our membership information packet, please check this box. □

**Shipping Information**

**Ship to:** ______________________________
______________________________
______________________________
______________________________

Please allow 2 – 3 weeks for delivery. Overseas orders are shipped via air printed matter. Please mail or fax this order form with payment to:

USENIX Association • 2560 Ninth Street • Suite 215 • Berkeley, CA • 94710 • TEL 510-528-8649 • FAX 510-548-5738

# LOCAL USER GROUPS

The Association will support local user groups by doing a mailing to assist in the formation of a new group and publishing information on local groups in *;login:*. At least one member of the group must be a current member of the Association. Send additions and corrections to: <*login@usenix.org*>.

## California

### Fresno:

The Central California UNIX Users Group consists of a uucp-based electronic mailing list to which members may post questions or information. For connection information:

- Educational and governmental institutions:
  Brent Auernheimer
  (209) 278-2573,
  <*brent@CSUFresno.edu*> or
  <*csufres!brent*>
- Commercial institutions or individuals:
- Gordon Crumal
  (209) 251-2648
  <*csufres!gordon*>

### Orange County

Meets the 2nd Monday of each month

- UNIX Users Association of Southern California
  Dave Close
  (714) 434-7359
  <*dhclose@alumni.caltech.edu*>
  New Horizons Computer Learning Center
  1231 E. Dyer Rd., Suite 140
  Santa Ana, CA 92705
  (714) 438-9440

## Colorado

### Boulder

Meets monthly at different sites. For meeting schedule, send email to <*fruug-info@fruug.org*>.

- Front Range UNIX Users Group
  Lone Eagle Systems Inc.
  636 Arapahoe #10
  Boulder, CO 80302
  Steve Gaede
  (303) 444-9114
  <*gaede@fruug.org*>

## Washington, D.C.

Meets 1st Tuesday of each month.

- Washington Area UNIX Users Group
  9811 Mallard Drive
  Laurel, MD 20708
  Alan Fedder
  (301) 953-3626

## Florida

### Coral Springs:

- S. Shaw McQuinn
  (305) 344-8686
  8557 W. Sample Road
  Coral Springs, FL 33065

### Melbourne:

Meets the 3rd Monday of every month.

- Space Coast UNIX User's Group
  Steve Lindsey
  (407) 242-4766
  <*lindsey@vnet.ibm.com*>

### Orlando:

Meets the 3rd Thursday of each month.

- Central Florida UNIX Users Group Mikel Manitius
  (407) 444-8448
  <*mikel@aaa.com*>

### Western:

Meets 1st Thursday of each month.

- Florida West Coast UNIX Users Group
  Richard Martino (813) 536-1776
  Tony Becker (813) 799-1836
  <*mcrsys!tony*>
  Ed Gallizzi, Ph.D. (813) 864-8272
  <*e.gallizzi@compmail.com*>
  Jay Ts (813) 979-9169
  <*uunet!pdn!tscs!metran!jan*>
  Dave Lewis (407)242-4372
  <*dhl@ccd.harris.com*>

## Georgia

### Atlanta:

Meets on the 1st Monday of each month in White Hall, Emory University.

- Atlanta UNIX Users Group
  P.O. Box 12241
  Atlanta, GA 30355-2241
  Mark Landry (404) 365-8108

## Kansas or Missouri

Meets on 2nd Tuesday of each month.

- Kansas City UNIX Users Group (KCUUG)
  P.O. Box 412622
  Kansas City, MO 64141
  (816) 891-1093
  <*richj@northcs.cps.com*>

## Michigan

### Detroit/Ann Arbor

Meets on the 2nd Thursday of each month in Ann Arbor.

- Southeastern Michigan Sun Local Users Group and Nameless UNIX Users Group
  Steve Simmons office:
  (313)769-4086
  home: (313) 426-8981
  <*scs@lokkur.dexter.mi.us*>

## Minnesota

### Minneapolis/St. Paul:

Meets the 1st Wednesday of each month.

- UNIX Users of Minnesota
  17130 Jordan Court
  Lakeville, MN 55044
  Robert A. Monio
  (612) 220-2427
  <*pnessutt@dmshq.mn.org*>

## Missouri

### St. Louis:

- St. Louis UNIX Users Group
  P..O. Box 2182 St. Louis, MO 63158
  Terry Linhardt
  (314) 772-4762
  <*uunet!jgaltstl!terry*>

## Nebraska

### Omaha: Meets monthly.

- /usr/group/nebraska
  P.O. Box 31012
  Omaha, NE 68132
  Phillip Allendorfer
  (402) 423-1400

## New England

### Northern:

Meets monthly at different sites.

- Peter Schmitt 603) 646-2085
  Kiewit Computation Center
  Dartmouth College Hanover, NH 03755
  <peter.schmitt@dartmouth.edu>

## New Jersey

### Princeton:

Meets monthly.

- Princeton UNIX Users Group
  Mercer County Community College
  1200 Old Trenton Road
  Trenton, NJ 08690
  Peter J. Holsberg (609) 586-4800
  *<mccc!pjh>*

## New Mexico

### Albuquerque:

ASIGUNIX meets every 3rd Wednesday of each month.

- Phil Hortz 505/275-0466.

## New York

### New York City:

Meets every other month in Manhattan.

- Unigroup of New York City
  G.P.O. Box 1931
  New York, NY 10116
  *<unigroup@murphy.com>*
  Bob Young (212) 490-8470

## Oklahoma

### Tulsa:

Meets 2nd Wednesday of each month.

- Yulsa UNIX Users Group, $USR
  Stan Mason (918) 560-5329
  *<tulsix!smason@drd.com>*
  Mark Lawrence (918) 743-3013
  *<mark@drd.com>*

## Texas

### Austin:

Meets 3rd Thursday of each month.

- Capital Area Central Texas UNIX Society (CACTUS)
  P.O. Box 9786
  Austin, TX 78766-9786
  Tom Painter (512) 258-7321
  *<president@cactus.org>*

### Dallas/Fort Worth:

Meets the 1st Thursday of each month.

- Dallas/Fort Worth UNIX Users Group
  P.O. Box 867405
  Plano, TX 75086
  Evan Brown (214) 519-3577
  *<evbrown@dsccc.com>*

### Houston:

Meets 3rd Tuesday of each month.

- Houston UNIX Users Group (Hounix) answering machine
  (713) 684-6590
  Bob Marcum, President
  (713) 626-4100
  Chuck Bentley, Vice-president
  (713) 789-8928
  *<chuckb@hounix.uucp>*

## Washington

### Seattle:

Meets monthly.

- Seattle UNIX Group Membership Info.
  Bill Campbell (206) 947-5591
  6641 East Mercer
  Mercer Island, WA 98040-0820
  *<bill@celestial.com>*

## Canada

### Manitoba:

Meets 2nd Tuesday of each month.

- Manitoba UNIX User Group
  (MUUG) P.O. Box 130
  St. Boniface Winnipeg,
  MB R2H 3B4
  Bary Finch, President
  (204) 934-2723
  *<info@muug.mb.ca>*

### Ottawa:

- The Ottawa Carleton UNIX Users Group
  D.J. Blackwood
  (613)957-9305
  *<dave@revcan.rct.ca>*

### Toronto:

- 143 Baronwood Court
  Brampton, Ontario
  Canada L6V 3H8
  Evan Leibovitch
  (416) 452-0504
  *<evan@telly.on.ca>*

### Quebec:

Meetings first Wednesday every 3rd month.

- Administrateurs de Systeme UNIX du Quebec (ASUQ)
  Universite de Montreal, Dept. IRO
  C.P. 6128, Succ. Centre-Ville
  Montreal, Quebec, Canada, H3C 3J7
  Tel: (514) 343-7480
  Fax: (514) 343-5834

## System Administration Groups

### Back Bay LISA (BBLISA)

New England forum covering all aspects of system and network administration, for large and small installations. Meets monthly, at MIT in Cambridge, MA.
For information, contact:

- J. R. Oldroyd (617)227-563
  *<jr@opal.com>*
- Mailing list subscription:
  *<requests:bblisaquest@cs.umb.edu>*
- Mailing list postings:
  *<bblisa@cs.umb.edu>*
- For current calendar of events:
  finger *<bblisa@cs.umb.edu>*

### Bay LISA

Meets 3rd Thursday of each month in Mountain View, CA For more information, please contact:
*<baylisa-info@baylisa.org>* or

- Bryan McDonald,
  BayLISA President
  *<bigmac@baylisa.org>*
  P.O. Box 64369
  Sunnyvale CA, 94088-4369
  (415) 859-3246

### $GROUPNAME (New Jersey)

$GROUPNAME is an organization in New Jersey formed to facilitate information exchange pertaining to the field of UNIX system administration. For more information, send email to:
*Majordomo@Warren. MENTORG.COM* or
Tom Limoncelli
*<tom_limoncelli@warren.mentorg.com>*

### New York Systems Administrators (NYSA)

Meets 2nd Monday of each month.

- *<nysa-request@esm.com>*
  914/472-3635

### North Carolina System Administrators Group

The North Carolina System Administrators Group meets on the 2nd Monday each month around the Research Triangle Park area.

- Amy Kreiling (919) 962-1843
  *<kreiling@cs.unc.edu>*
- William E. Howell (919) 962-1717
  *<howell@cs.unc.edu>*

**ACM:**
Association for Computing Machinery

**AFUU:**
Association of French UNIX Users

**ASPLOS:**
Architectural Support for Programming Languages and Operating Systems

**AUUG:**
Australian UNIX Systems Users Group

**COOTS:**
Conference on Object-Oriented Technologies

**DECUS:**
Digital Equipment Computer Users Society

**EurOpen:**
European Forum for Open Systems

**GURU:**
Romanian UNIX User Group

**GUUG:**
German UNIX Users Group

**IEEE:** Institute of Electrical and Electronics Engineers

**IETF:**
Internet Engineering Task Force

**Interex:**
International Association of Hewlett-Packard Computer Users

**JUS:**
Japan UNIX Society

**LISA:**
USENIX/SAGE Systems Administration Conference

**OOPSLA:**
Object-oriented Programming Systems, Languages and Applications

**OSDI:**
Symposium on Operating Systems Design & Implementation

**SANS:**
System Administration, Networking & Security

**SOSP:**
Symposium on Operating Systems Principles

**SUG:**
Sun User Group

**SUUG:**
Society of Russia UNIX Users Group

**UKUUG:**
United Kingdom UNIX Systems Users Group

**UniForum:**
International Association of UNIX and Open Systems Professionals

# CALENDAR OF EVENTS

This is a combined calendar of conferences, symposia, and standards meetings. If you have a event that you wish to publicize, please contact *<login@usenix.org>*.

* = events sponsored by the USENIX Association.

## 1995

### February

| | |
|---|---|
| 8-10 | Advanced UNIX & Internet Security, EurOpen, London, England |

### March

| | |
|---|---|
| 1-3 | GUUG Spring Conference, Cologne, Germany |
| 13-17 | UniForum, Dallas, TX |
| 21-23 | AFUU Unix '95 Conference Paris, France |
| 27-31 | NetWorld + Interop 95, Las Vegas, NV |
| 28-31 | Fifth Conference on Computers, Freedom and Privacy, Burlingame, CA |

### April

| | |
|---|---|
| 3 -7 | IETF, Danvers, MA |
| 10-11* | 2nd Mobile and Location–Independent Computing, Ann Arbor, MI |
| 10-14 | IEEE 1003 |
| 10-14 | 3rd International World Wide Web Conference, Darmstadt, Germany |
| 24-29 * | SANS IV, Washington, DC |

### May

| | |
|---|---|
| 4 - 5 | IEEE 5th Workshop on Hot Topics in Operating Systems Orcas Island, WA |
| 13-19 | DECUS, Atlanta, GA |
| 17-20 | UniForum NZ, Masterton, New Zealand |
| 29-J2 | NetWorld + Interop 95, Frankfurt, Germany |

### June

| | |
|---|---|
| 5 - 7 * | UNIX Security, Salt Lake City, UT |
| 26-29* | COOTS, Monterey, CA |

### July

| | |
|---|---|
| 6-11* | Tcl/Tk Workshop, Toronto, Canada |
| 10-14 | IEEE 1003 |
| 17-21 | IETF, Stockholm, Sweden |
| 17-21 | NetWorld + Interop 95, Tokyo, Japan |

### August

| | |
|---|---|
| 6-11 | ACM Siggraph, Los Angeles, CA |
| 14-18 | Interex 95, Toronto, Canada |

### September

| | |
|---|---|
| 12-14 | GUUG Annual Conference, Weisbaden, Germnany |
| 18-22* | LISA '95, Monterey, CA |
| 19-21 | UNIX Expo, New York City |
| 25-29 | NetWorld + Interop 95, Atlanta, GA |

### October

| | |
|---|---|
| 9-13 | IEEE 1003 |
| 15-19 | OOPSLA, Austin, TX |

### November

| | |
|---|---|
| 2- 8 | DECUS, San Francisco, CA |
| 6-10 | NetWorld + Interop 95, Paris |

### December

| | |
|---|---|
| 3-6 | SOSP, Colorado |
| 4-8 | IETF, Dallas, TX |
| 4-8 | IEEE Supercomputing, San Diego, CA |

## 1996

### January

| | |
|---|---|
| 22-26* | USENIX, San Diego, CA |

### February

| | |
|---|---|
| 14-16 | UniForum, San Francisco, CA |

### May

| | |
|---|---|
| 18-24 | DECUS, Orlando, FL |

### August

| | |
|---|---|
| 4 - 8 | Interex 96, San Diego, CA |

### September

| | |
|---|---|
| | AUUG, Melbourne, Australia |

### October

| | |
|---|---|
| 8-10 | UNIX Expo, New York City |

### November

| | |
|---|---|
| 16-22 | DECUS, Anaheim, CA |

### December

| | |
|---|---|
| | JUS UNIX Fair, Tokyo, Japan |

# Integrate Macs with Sun, SGI, & HP

## K-AShare/K-FS

### Share resources seamlessly

Macintosh® and UNIX® workstation users share files effortlessly. K–AShare™ brings UNIX-resident files to Macintosh computers; K-FS™ brings Macintosh files to UNIX workstations. Cross-platform applications like Frame, Photoshop and Illustrator can read and write the same files from either system.

### Better than an AppleShare file server

UNIX workstations running K-AShare provide higher performance than dedicated AppleShare servers while allowing Mac users to continue using the familiar Mac interface.

And, the UNIX workstation simultaneously gets other important jobs done for you.

## K-Spool

### More printing options

Macintoshes and UNIX systems talk to all PostScript printers and image setters regardless of how they're connected. Each system continues to use its own familiar printing interface.

### Increased productivity

Macs finish printing more quickly, and UNIX workstations have access to many more printing devices. K-Spool also brings the power of SGI's Impressario™ to your Macs.

Introducing

## FullPress

### Integrated prepress management system

FullPress™ is the workflow solution to prepress file sharing, print spooling and OPI, affording increased productivity in prepress operations where multiple Macs are used to produce complex pieces and imagesetting.

inet
560 9th St., #312
erkeley, CA 94710
l: 510 845 0555
x: 510 644 2680
mail: uinfo@xinet.com

;login:

**USENIX Association**
**2560 Ninth Street**
**Suite 215**
**Berkeley, CA 94710**

**POSTMASTER:**
SEND ADDRESS CHANGES
TO ***;login:***
USENIX ASSOCIATION
2560 NINTH STREET
SUITE 215
BERKELEY, CA 94710

# Many thanks to our Corporate Sponsors!

Hewlett-Packard • Apollo Computer, Inc. • BBN • Timex • Sealand Svc., Inc. • CCR-P • Siemens Corp. Research • New Jersey Medical Underwriters Inc. • ProLogic Corporation • Lehman Brothers • The Sword Group Inc. • Hazeltin Corporation • Brookhaven National Lab. • Northern Telecom • Alcoa Technical Center • Defense Industrial Supply Center • American Chemical Society Library • National Institutes of Health • C & P Telephone Company • RMS Associates • NCI-Frederich Cancer R&D Center • PRC, Inc. • Teledyne Brown Eng. • USA Topographical Engineering Center • Philip Morris, USA • MCNC • E. Systems • OCLC Library • Moen Inc. • State of Indiana • Eli Lilly $ Company • Cimlinc, Inc. • Wisconsin Department of Transportation • UNISYS • Cray Research, Inc. • Motorola Inc. • IEX Corporation • Ericsson Network Systems • Halliburton Company • Rosetta, Inc. • Baker Hughes Inteq • EG&G Idaho Inc. • 3M Health Information Systems • Mentat, Inc. • Advanced Computer Support Center • Rand Corporation • Chevron InformationTechnologies • Siemens Pacesetter, Inc. • Peregrine Systems , Inc. • Qualcom, Inc. • Speech Technology Lab • Sterling Software US, Inc. • Sun Microsystems, Inc. • Genentech, Inc. • Advanced Micro Devices • Amdahl Corporation • Wollongong Group • Fuji • Xerox Palo Alto Lab • Sun Microsystems Labs. • Montage Software Inc. • Capital Market Technologies • Connect Inc. • Apple Computer, Inc. • Intel Corporation • O'Reilly & Associates, Inc. • Hewlett Packard Technical Information Center • John Fluke Mfg. Co. Inc. • Orbital Communications • Message Handling System Pty. Ltd. • The Land Information Centre Department of Lands • Fulcrum Consulting • IIASA • Sobeco Group Inc. • Sun Microsystems • Institut de recherche d'Hydro-Quebec • Comm. Security Establishment • SCO Canada, Inc. • Toronto Stock Exchange • Mortice Kern Systems, Inc. • City of Calgary • Sandwell, Inc. Data Systems Division • UNI-C • ICL Corporate Systems Division • BULL • Institut Pasteur • Apple Computer Europe • Terminaux Postes Travail ICI IMA • EDS Electronic Data Systems • ECRC • SSO UNIX Software Support • Altos India Ltd. • HCL Hewlett-Packard Ltd. • Motorola Ireland Ltd. • Nippon • Unisoft Corporation • Software Research Center, Ricoh Co. Ltd. • Surigiken Co., Ltd. • Access Co., Ltd • Software Research Association • NEC Corporation • NTT Data Communications Systems Corp. • CSK Corporation • Fuji • Xerox, Service Core Development Department. • Hitachi, Ltd. • OKI Technosystems Laboratory • National Computerization Agency • Nederlandse Philips Bedrijven BV • Delft Hydraulics • NZ Apple & Pear Marketing Board • Norsk Regnesentral • ENEA Data AB • UPSYS AB • DynaSoft, Dynamic Software AB • Ericsson Telecom AB • CERN • IBM Research Division • Union Bank of Switzerland • Goldman Sachs • Swiss Bank Corporation • Canon Research Centre Europe • Richard Shooter • AEA Technology - Harwell • Digital Equipment Corporation • P.O. Research Centre • Salomon Bros. Intl. Ltd.